

OOP PHP

Nur Muhammad Abdul Falaq

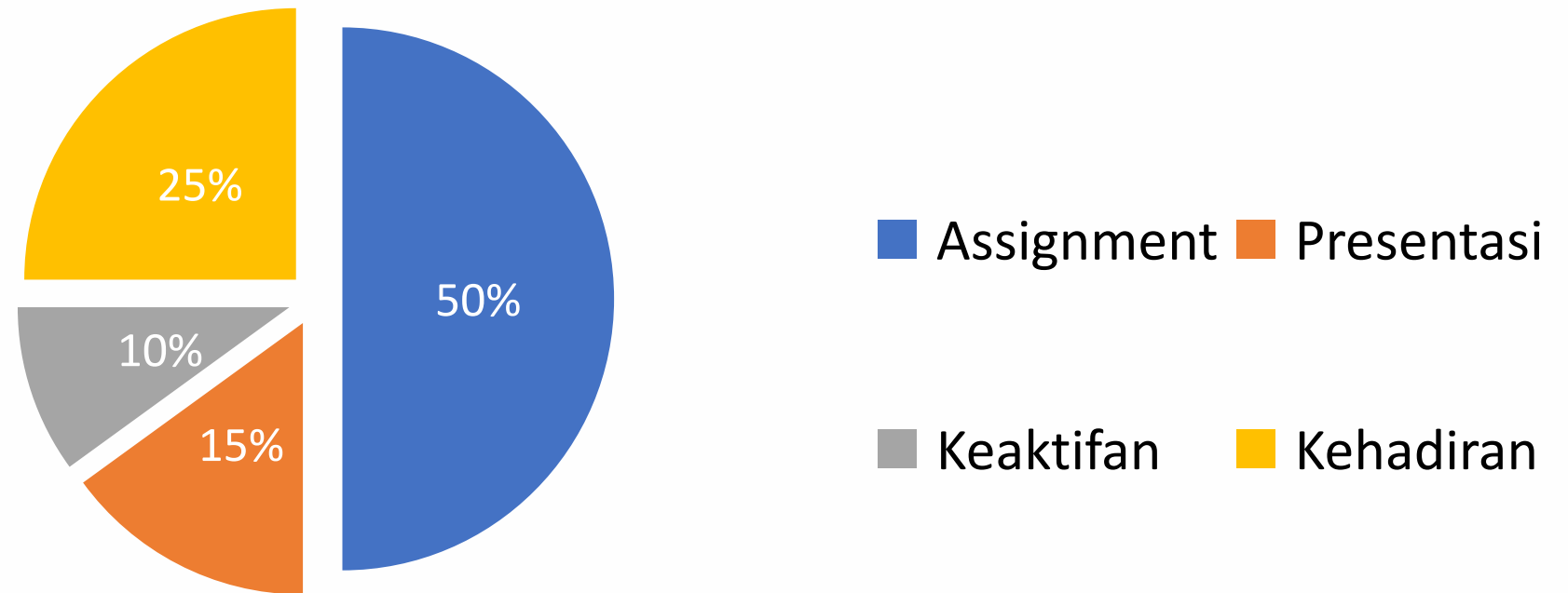


PINTAR



Penilaian

Komponen Penilaian



Scope

- OOP
- Class
- Object
- Properties
- Function
- this key
- Contruction
- Inheritance

Referensi

- https://www.youtube.com/watch?v=_P2t0lCzU-Q&t=492s

OOP PHP

- Object Oriented Programming adalah sudut pandang bahasa pemrograman yang berkonsep “objek”
- Ada banyak sudut pandang bahasa pemrograman, namun OOP adalah yang sangat populer saat ini.
- Ada beberapa istilah yang perlu dimengerti dalam OOP, yaitu: Object dan Class

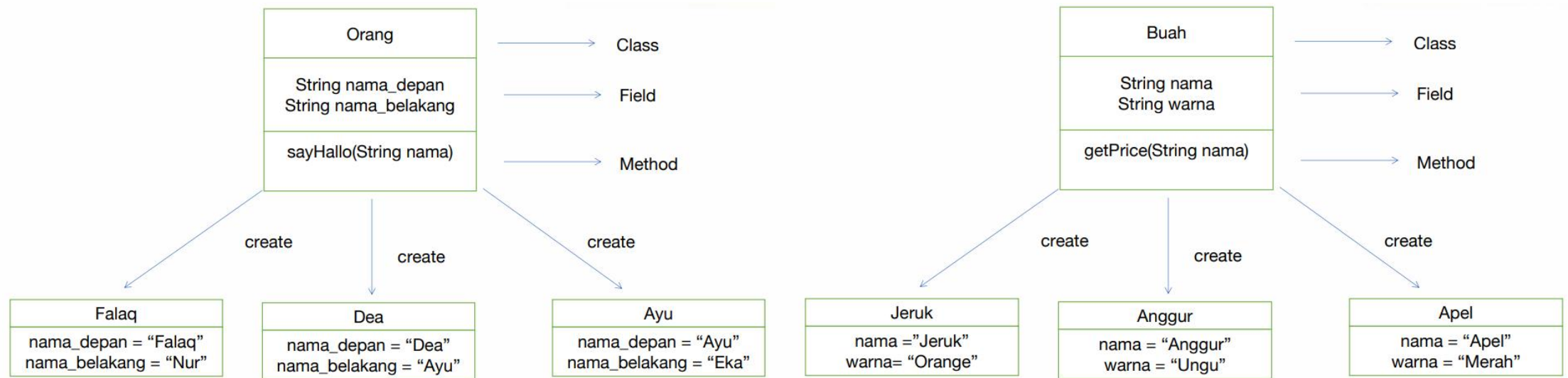
Apa itu Object?

- Object adalah data yang berisi field / properties / attributes dan method / function / behavior

Apa itu Class?

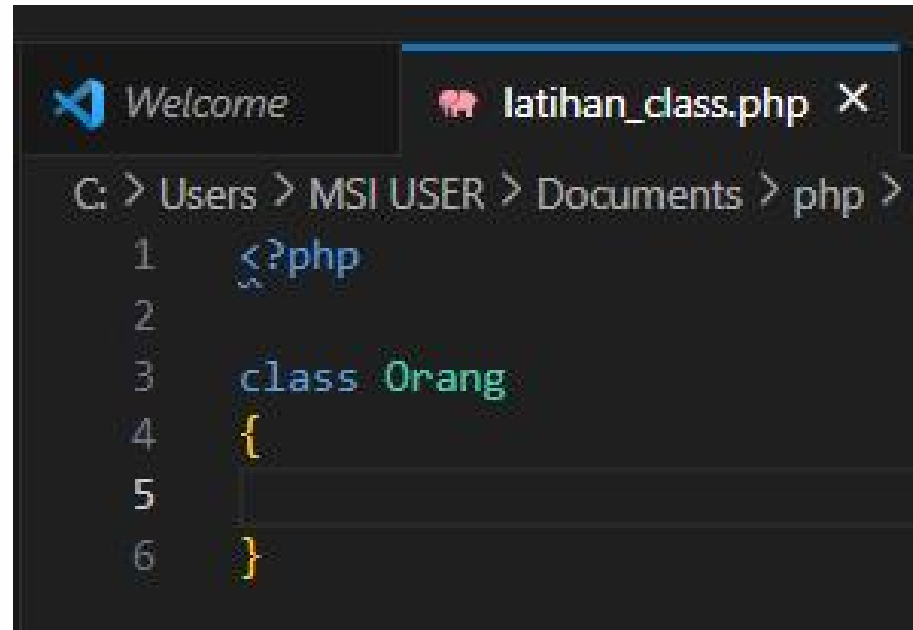
- Class adalah blueprint, prototype atau cetakan untuk membuat Object
- Class berisikan deklarasi semua properties dan functions yang dimiliki oleh Object
- Setiap Object selalu dibuat dari Class
- Dan sebuah Class bisa membuat Object tanpa batas

OOP PHP



Membuat Class

- Untuk membuat class, kita bisa menggunakan kata kunci class
- Penamaan class biasa menggunakan format CamelCase

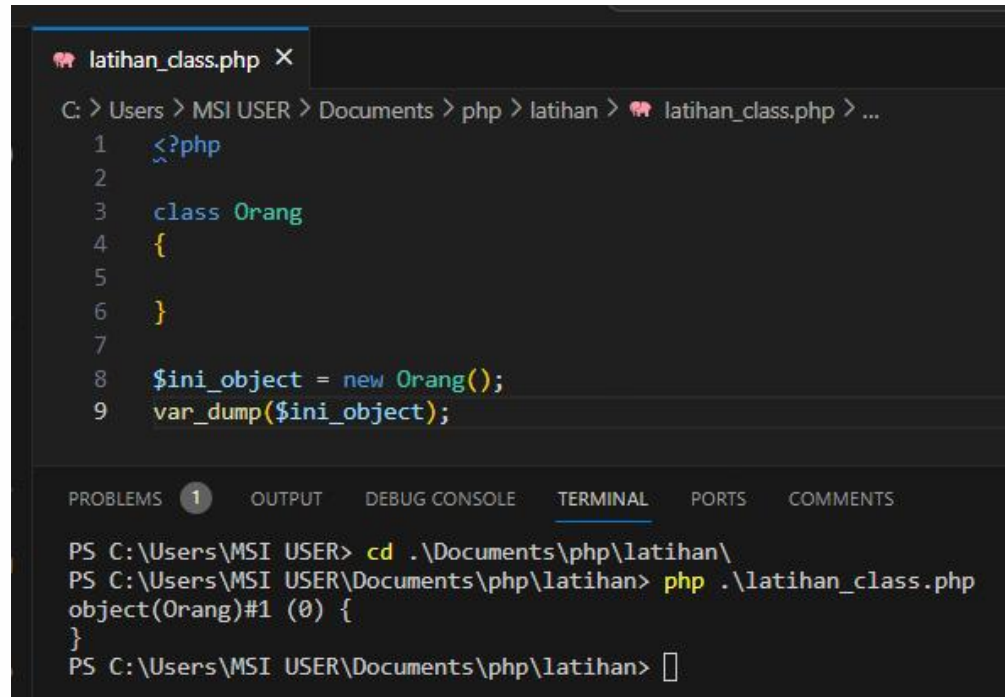


```
1 <?php
2
3 class Orang
4 {
5
6 }
```

PascalCase

Membuat Object

- Object adalah hasil instansiasi dari sebuah class
- Untuk membuat object kita bisa menggunakan kata kunci new, dan diikuti dengan nama Class dan kurung ()



```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  class Orang
4  {
5
6  }
7
8  $ini_object = new Orang();
9  var_dump($ini_object);

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
PS C:\Users\MSI USER> cd .\Documents\php\latihan\
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
object(Orang)#1 (0) {
}
PS C:\Users\MSI USER\Documents\php\latihan>
```

Properties

- Fields / Properties / Attributes adalah data yang bisa kita sisipkan di dalam Object
- Namun sebelum kita bisa memasukkan data di fields, kita harus mendeklarasikan data apa aja yang dimiliki object tersebut di dalam deklarasi class-nya
- Membuat field sama seperti membuat variable, namun ditempatkan di block class, namun diawali dengan kata kunci var

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > Orang >
1  <?php
2
3  class Orang
4  {
5      var $nama;
6      var $alamat;
7      var $umur;
8  }
9
10 $ini_object = new Orang();
11 var_dump($ini_object);

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
object(Orang)#1 (3) {
    ["nama"]=>
    NULL
    ["alamat"]=>
    NULL
    ["umur"]=>
    NULL
}
PS C:\Users\MSI USER\Documents\php\latihan>
```


Manipulasi Properties

- Fields yang ada di object, bisa kita manipulasi.
- Untuk memanipulasi data field, sama seperti cara pada variable
- Untuk mengakses field, kita butuh kata kunci -> setelah nama object dan diikuti nama fields nya

```
10 $ini_object = new Orang();  
11 $ini_object->nama = 'falaq';  
12 $ini_object->alamat = 'jogja';  
13 $ini_object->umur = '17 tahun';  
14  
15 echo "Nama : $ini_object->nama \n";  
16 echo "Alamat : $ini_object->alamat \n";  
17 echo "Umur : $ini_object->umur \n";
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php  
Nama : falaq  
Alamat : jogja  
Umur : 17 tahun  
PS C:\Users\MSI USER\Documents\php\latihan> 
```

Default Properti Value

- Sama seperti variable, di properties juga kita bisa langsung mengisi value nya
- Ini mirip seperti default value, jadi jika tidak diubah di object, maka properties akan memiliki value tersebut

C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...

```
1  <?php
2
3  class Orang
4  {
5      var $nama = "nico";
6      var $alamat = "Jakarta";
7      var $umur = "15 tahun";
8  }
9
10 $ini_object = new Orang();
11 $ini_object->nama = 'falaq';
12
13 echo "Nama : $ini_object->nama \n";
14 echo "Alamat : $ini_object->alamat \n";
15 echo "Umur : $ini_object->umur \n";
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
Nama : falaq
Alamat : Jakarta
Umur : 15 tahun
PS C:\Users\MSI USER\Documents\php\latihan> 
```

Function

- Selain menambahkan properties, kita juga bisa menambahkan function ke object
- Cara dengan mendeklarasikan function tersebut di dalam block class
- Sama seperti function biasanya, kita juga bisa menambahkan return value dan parameter
- Untuk mengakses function tersebut, kita bisa menggunakan tanda -> dan diikuti dengan nama method nya. Sama seperti mengakses properties

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  class Orang
4  {
5      var $nama = "nico";
6      var $alamat = "Jakarta";
7      var $umur = "15 tahun";
8
9      function sayHello(string $nama) {
10         echo "Halo nama saya $nama";
11     }
12 }
13
14 $ini_object = new Orang();
15 $ini_object->sayHello('falaq');
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
Halo nama saya falaq
```

This Keyword

- Saat kita membuat kode di dalam function di dalam class, kita bisa menggunakan kata kunci this untuk mengakses object saat ini
- Misal kadang kita butuh mengakses properties atau function lain di class yang sama

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > Orang > s
1  <?php
2
3  class Orang
4  {
5      var $nama = "nico";
6      var $alamat = "Jakarta";
7      var $umur = "15 tahun";
8
9      function sayHello(?string $nama) {
10         if(is_null($nama)){
11             echo "Halo nama saya $this->nama";
12         }else{
13             echo "Halo nama saya $nama bukan $this->nama";
14         }
15     }
16 }
17
18 $ini_object = new Orang();
19 $ini_object->sayHello('falaq');
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
Halo nama saya falaq bukan nico
PS C:\Users\MSI USER\Documents\php\latihan> 
```

Constructor

- Saat kita membuat Object, maka kita seperti memanggil sebuah function, karena kita menggunakan kurung ()
- Di dalam class PHP, kita bisa membuat constructor, constructor adalah function yang akan dipanggil saat pertama kali Object dibuat.
- Mirip seperti di function, kita bisa memberi parameter pada constructor
- Nama constructor di PHP haruslah __construct()

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  class Orang
4  {
5      var $nama = "nico";
6      var $alamat = "Jakarta";
7      var $umur = "15 tahun";
8
9      public function __construct($nama, $alamat, $umur)
10     {
11         $this->nama = $nama;
12         $this->alamat = $alamat;
13         $this->umur = $umur;
14     }
15
16     function sayHello(?string $nama) {
17         if(is_null($nama)){
18             echo "Halo nama saya $this->nama";
19         }else{
20             echo "Halo nama saya $nama bukan $this->nama";
21         }
22     }
23 }
24
25 $ini_object = new Orang('falaq', 'jogja', '17 tahun');
26 $ini_object->sayHello('dadang');
```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
Halo nama saya dadang bukan falaq
PS C:\Users\MSI USER\Documents\php\latihan>
```


Inheritance

- Inheritance atau pewarisan adalah kemampuan untuk menurunkan sebuah class ke class lain
- Dalam artian, kita bisa membuat class Parent dan class Child
- Class Child, hanya bisa punya satu class Parent, namun satu class Parent bisa punya banyak class Child
- Saat sebuah class diturunkan, maka semua properties dan function yang ada di class Parent, secara otomatis akan dimiliki oleh class Child
- Untuk melakukan pewarisan, di class Child, kita harus menggunakan kata kunci extends lalu diikuti dengan nama class parent nya.

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  class BuahBuahan
4  {
5      var $batang = "berbatang kayu";
6      var $buah = "berbiji";
7      var $bunga = "berbunga sempurna";
8
9      function sayBuah($buah) {
10         echo "ini adalah buah $buah \n";
11     }
12 }
13
14 class Apple extends BuahBuahan
15 {
16 }
17
18
19 $buah = new Apple();
20 $buah->sayBuah('apel');
21 echo $buah->bunga;
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
ini adalah buah apel
berbunga sempurna
PS C:\Users\MSI USER\Documents\php\latihan>
```

```
latihan_class.php X
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  class BuahBuahan
4  {
5      var $batang = "berbatang kayu";
6      var $buah = "berbiji";
7      var $bunga = "berbunga sempurna";
8  }
9
10 class Apple extends BuahBuahan
11 {
12 }
13
14
15 $buah = new Apple();
16 echo $buah->bunga;
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

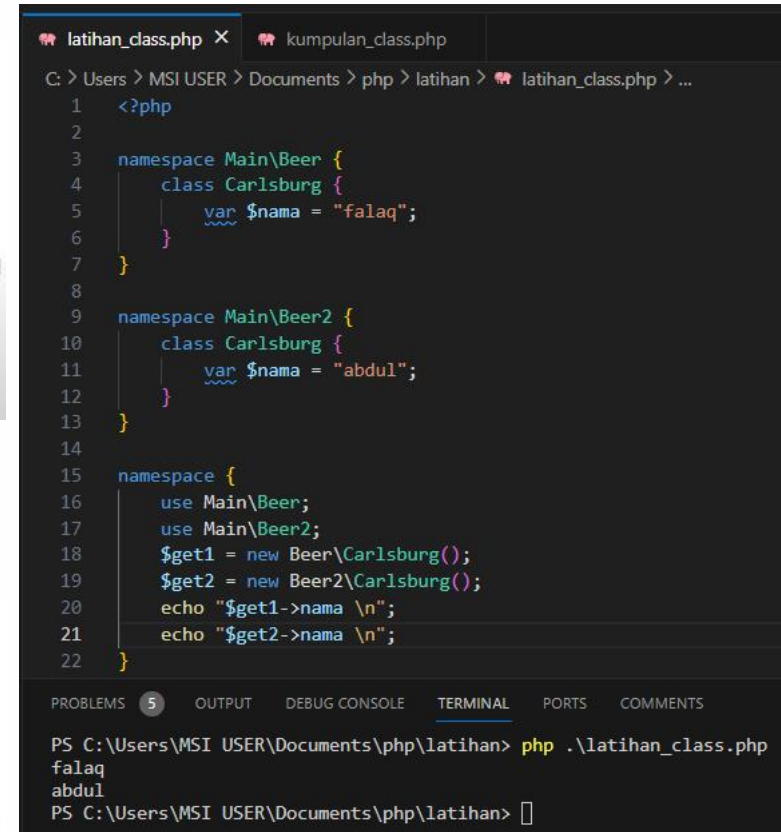
```
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
berbunga sempurna
PS C:\Users\MSI USER\Documents\php\latihan>
```

Namespace

- Saat kita membuat aplikasi, bisa dipastikan kita akan banyak sekali membuat class
- Jika class terlalu banyak, kadang akan menyulitkan kita untuk mencari atau mengklasifikasikan jenis-jenis class
- PHP memiliki fitur namespace, dimana kita bisa menyimpan class-class kita di dalam namespace
- Namespace bisa nested, dan jika kita ingin mengakses class yang terdapat di namespace, kita perlu menyebutkan nama namespace nya
- Namespace bagus ketika kita punya beberapa class yang sama, dengan menggunakan namespace nama class sama tidak akan menjadikan error di PHP

Membuat Namespace

- Untuk membuat namespace, kita bisa menggunakan kata kunci namespace
- Jika kita ingin membuat sub namespace, kita cukup gunakan karakter \ setelah namespace sebelumnya

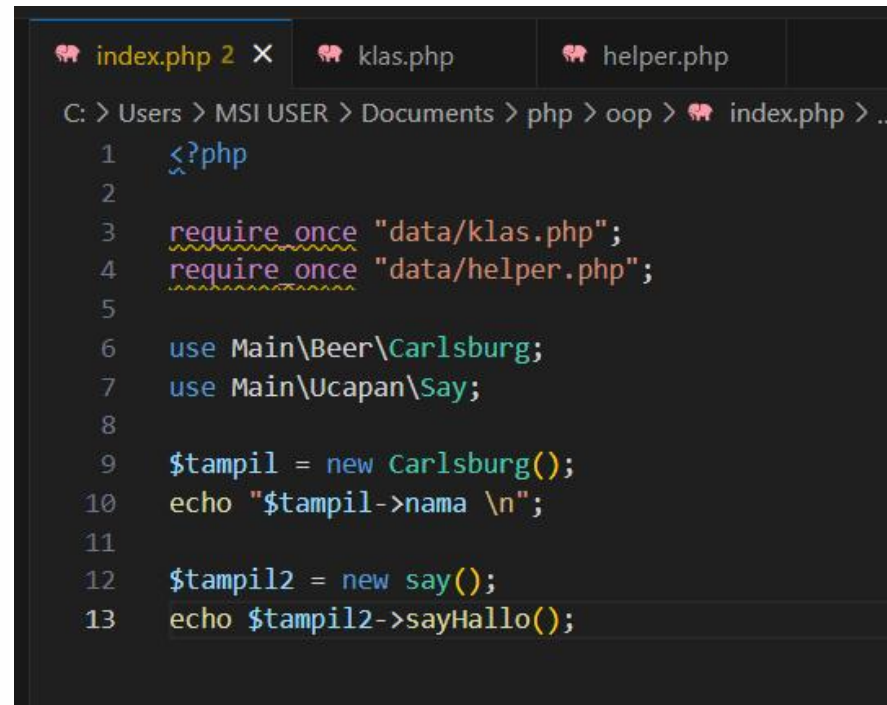


```
latihan_class.php X kumpulan_class.php
C: > Users > MSI USER > Documents > php > latihan > latihan_class.php > ...
1  <?php
2
3  namespace Main\Beer {
4      class Carlsburg {
5          var $nama = "falaq";
6      }
7  }
8
9  namespace Main\Beer2 {
10     class Carlsburg {
11         var $nama = "abdul";
12     }
13 }
14
15 namespace {
16     use Main\Beer;
17     use Main\Beer2;
18     $get1 = new Beer\Carlsburg();
19     $get2 = new Beer2\Carlsburg();
20     echo "$get1->nama \n";
21     echo "$get2->nama \n";
22 }

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
PS C:\Users\MSI USER\Documents\php\latihan> php .\latihan_class.php
falaq
abdul
PS C:\Users\MSI USER\Documents\php\latihan> 
```

Import

- Sebelumnya kita sudah tahu bahwa untuk menggunakan class, function atau constant di namespace kita perlu menyebutkan nama namespace nya di awal
- Jika terlalu sering menggunakan class, function atau constant yang sama, maka terlalu banyak duplikasi dengan menyebut namespace yang sama berkali-kali
- Hal ini bisa kita hindari dengan cara mengimport class, function atau constant tersebut dengan menggunakan kata kunci use



```
index.php 2 x  klas.php  helper.php
C: > Users > MSI USER > Documents > php > oop > index.php > ...
1  <?php
2
3  require_once "data/klas.php";
4  require_once "data/helper.php";
5
6  use Main\Beer\Carlsburg;
7  use Main\Ucapan\Say;
8
9  $stampil = new Carlsburg();
10 echo "$stampil->nama \n";
11
12 $stampil2 = new say();
13 echo $stampil2->sayHallo();
```


Alias

- Saat kita menggunakan use, artinya kita tidak perlu lagi menggunakan nama namespace diawal class ketika kita ingin membuat class tersebut
- Namun bagaimana jika kita ternyata nama class nya sama?
- Untungnya PHP memiliki fitur yang namanya alias
- Alias adalah kemampuan membuat nama lain dari class, function atau constant yang ada
- Kita bisa menggunakan kata kunci as setelah melakukan use

```
index.php 2 x  klas.php  helper.php
C: > Users > MSI USER > Documents > php > oop > index.php > ...
1  <?php
2
3  require_once "data/klas.php";
4  require_once "data/helper.php";
5
6  use Main\Beer\Carlsburg as gajah;
7  use Main\Ucapan\Say as ucap;
8
9  $stampil = new gajah();
10 echo "$stampil->nama \n";
11
12 $stampil2 = new ucap();
13 echo $stampil2->sayHallo();
```

Tugas ...

1. Silahkan coba latihan yang ada di materi
2. Silahkan kumpulkan di github dengan membuat branch baru, format branch yaitu pertemuan-14-{nama}
contoh: pertemuan-14-abdul-falaq

 **tugas**

