

Web Programming

#10 Object Oriented PHP



Herman Kabetta

Understanding Object-Oriented Programming

A style of coding in which related actions are grouped into classes to aid in creating more-compact, effective code.

```
def forall(s: Set, p: Int => Boolean): Boolean = {  
  def iter(a: Int): Boolean = {  
    if (contains(s,a) && !p(a)) false  
    else if (a > 1000) true  
    else iter(a + 1)  
  }  
  iter(-1000)  
}  
  
def exists(s: Set, p: Int => Boolean): Boolean = {  
  ...  
}  
  
def map(s: Set, f: Int => Int): Set = {  
  (b: Int) => exists(s, f(b))  
}
```

Reusable

Extensible

Manageable

Terminologi

- **Object** → dapat berupa Class atau Instances. Harus berasal dari entitas atau konsep dunia nyata.
- **Class** → template untuk membuat obyek.
- **Attribute/Property** → identitas unik dari obyek
- **Method** → fungsi untuk pengaksesan atribut atau tugas tertentu
- **Encapsulation** → menyembunyikan struktur data dan implementasi suatu class.
- **Inheritance** → merepresentasikan keterhubungan struktural antar class
- **Polymorphism** → kemampuan untuk merepresentasikan 2 bentuk obyek yang berbeda

Encapsulation

Mengatur aksesibilitas member (properti dan method)

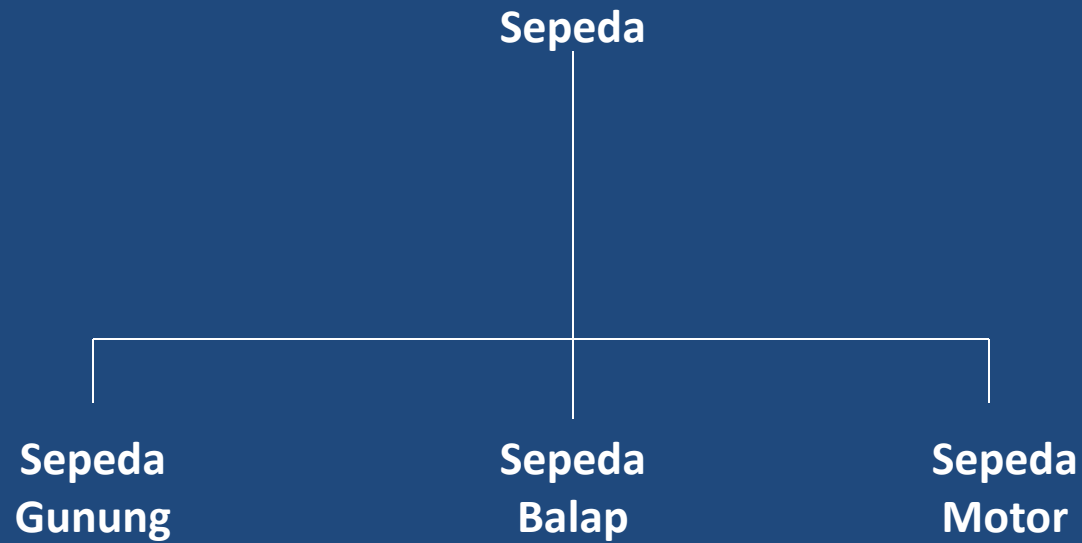
Public

Private

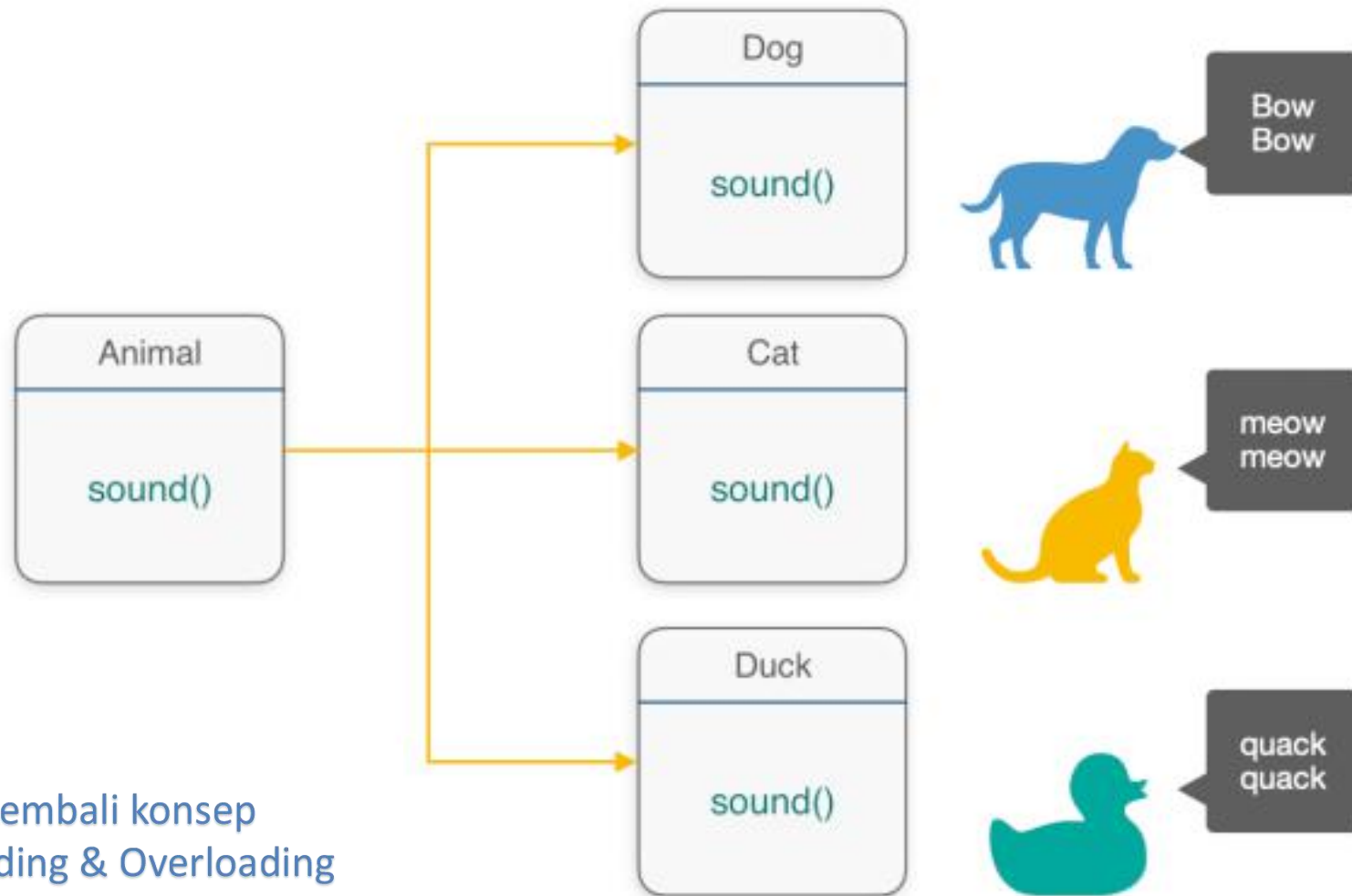
Protected



Inheritance



Polymorphism



Ingat kembali konsep
Overriding & Overloading

Structuring Class

```
<?php  
  
class MyClass  
{  
    // Class properties and methods go here  
}  
  
?>
```

After creating the class, a new class can be instantiated and stored in a variable using the “new” keyword:

```
$obj = new MyClass;
```

Membuat Class dan Object

latihan1.php

```
1  <?php
2
3  class Manusia
4  {
5      public $nama = "Nama saya Obi!";
6  }
7
8  $obj = new Manusia;
9  echo $obj->nama;
10
11  ?>
```


Membuat Method

```
1  <?php
2
3  class Manusia
4  {
5      public $nama;
6      public $umur;
7
8      function berbicara() {
9          echo "Hallo, nama saya $this->nama";
10     }
11 }
12
13 $obj = new Manusia();
14 $obj->nama = "Andara";
15 echo $obj->berbicara();
16
17 ?>
```

Membuat Method

latihan1.php

```
10  function statususia() {  
11      if ($this-> umur >= 17 )  
12          $status = 'Dewasa' ;  
13      else  
14          $status = 'Dibawah Umur';  
15      return $status;  
16  }
```

Tambahkan method diatas pada class Manusia

Instance Object

latihan1.php

```
19 $obj2 = new Manusia();  
20 $obj2->nama = 'Abimanyu';  
21 $obj2->umur = 10;  
22 $obj2->berbicara();  
23 echo "<br>";  
24 echo "saya ".$obj2->statususia();
```

Tambahkan kode (diluar class) untuk membuat objek baru

Encapsulation (Public)

latihan2.php

```
1  <?php
2  class mahasiswa {
3      public $nim;
4      public $nama;
5      public $nilai;
6
7      public function prosesNilai () {
8          echo "Mahasiswa dengan NIM $this->nim <br />";
9          echo "dengan Nama $this->nama <br />";
10         echo "Mendapatkan Nilai $this->nilai";
11     }
12 }
13
14 $objMhs=new mahasiswa();
15 $objMhs->nim = '122001321';
16 $objMhs->nama = 'Chepi Nurdiansyah';
17 $objMhs->nilai = 85;
18 $objMhs->prosesNilai();
19 ?>
```

Encapsulation (Private)

latihan2.php

```
3     private $nim;  
4     private $nama;  
5     private $nilai;
```

Encapsulation (Private)

latihan2.php

```
7  function setNim($x) {  
8      $this->nim=$x;  
9  }  
10 function setNama($x) {  
11     $this->nama=$x;  
12 }  
13 function setNilai($x) {  
14     $this->nilai=$x;  
15 }
```

Tambahkan method diatas pada class mahasiswa

Encapsulation (Private)

latihan2.php

```
24 $objMhs=new mahasiswa();  
25 $objMhs->setNim('122001321');  
26 $objMhs->setNama('Chepi Nurdiansyah');  
27 $objMhs->setNilai(85);  
28 $objMhs->prosesNilai();
```

Constructor dan Destructor

Method khusus yang otomatis dijalankan ketika object terbentuk atau dihapus



Constructor & Destructor

latihan3.php

```
2  class orang{
3      private $nama;
4
5      function __construct($nama) {
6          $this->nama=$nama;
7          echo "Constructor: $this->nama dibuat<br>";
8      }
9
10     function berbicara() {
11         echo "Hallo. Nama Saya adalah ".$this->nama."<br />";
12     }
13
14     function __destruct() {
15         echo "Destructor: $this->nama dihapus<br />";
16     }
17 }
```

Constructor & Destructor

latihan3.php

```
19  $orang1=new Orang("Orang 1");  
20  $orang1->berbicara();  
21  $orang2=new Orang("Orang 2");  
22  $orang2->berbicara();
```

Constructor & Destructor

latihan4.php

```
2  class Nilai{  
3      private $tugas;  
4      private $uts;  
5      private $uas;  
6  
7  }
```

Constructor & Destructor

latihan4.php

```
7 // constructor pemberian nilai awal nilai dengan 0
8 function __construct() {
9     $this->tugas=0;
10    $this->uts=0;
11    $this->uas=0;
12    echo "Constructor : Nilai Properti Tugas, ";
13    echo "UTS dan UAS diset 0.";
14    echo "<br /><br />";
15 }
```

Constructor & Destructor

latihan4.php

```
17 // fungsi untuk menset nilai tugas, diset dari 0 s.d 100
18 function settugas($nilai){
19     if(($nilai<=100)&&($nilai>=0))
20         $this->tugas=$nilai;
21 }
22
23 // fungsi untuk menset nilai uts, diset dari 0 s.d 100
24 function setuts($nilai){
25     if(($nilai<=100)&&($nilai>=0))
26         $this->uts=$nilai;
27 }
28
29 // fungsi untuk menset nilai uas, diset dari 0 s.d 100
30 function setuas($nilai){
31     if(($nilai<=100)&&($nilai>=0))
32         $this->uas=$nilai;
33 }
```

Constructor & Destructor

latihan4.php

```
35 // fungsi untuk mengambil nilai isian properti tugas
36 function gettugas(){
37     return $this->tugas;
38 }
39
40 //fungsi mengambil nilai isian properti uts
41 function getuts(){
42     return $this->uts;
43 }
44
45 // fungsi untuk mengambil nilai isian properti uas
46 function getuas(){
47     return $this->uas;
48 }
49
50 // fungsi untuk menghitung nilai akhir
51 function getNA(){
52     $nilaiakhir=0.2*$this->tugas+0.3*$this->uts+0.5*$this->uas;
53     return $nilaiakhir;
54 }
```

Constructor & Destructor

latihan4.php

```
56 // fungsi utk menampilkan nilai tugas, uts, uas dan nilai akhir
57 function tampil() {
58     echo "Nilai Tugas : ".$this->tugas.
59     ", Nilai UTS : ".$this->uts.
60     ", Nilai UAS : ".$this->uas.
61     ", Nilai akhir : ".$this->getNA()." <br />";
62 }
63
```

```
64 // destructor untuk menghapus objek dari memory
65 function __destruct() {
66     echo "<br /><br />";
67     echo "Destructor: Nilai Tugas, UTS dan UAS dihapus dari Memory<br />";
68 }
```

Constructor & Destructor

latihan4.php

```
72 $nilai=new Nilai();  
73 $nilai->settugas(80); // Set nilai tugas  
74 echo "Nilai Tugas sekarang : ".$nilai->gettugas();  
75 echo "<br />";  
76 $nilai->setuts(60); // Set nilai uts  
77 $nilai->setuas(90); // Set nilai uas  
78 $nilai->tampil();
```

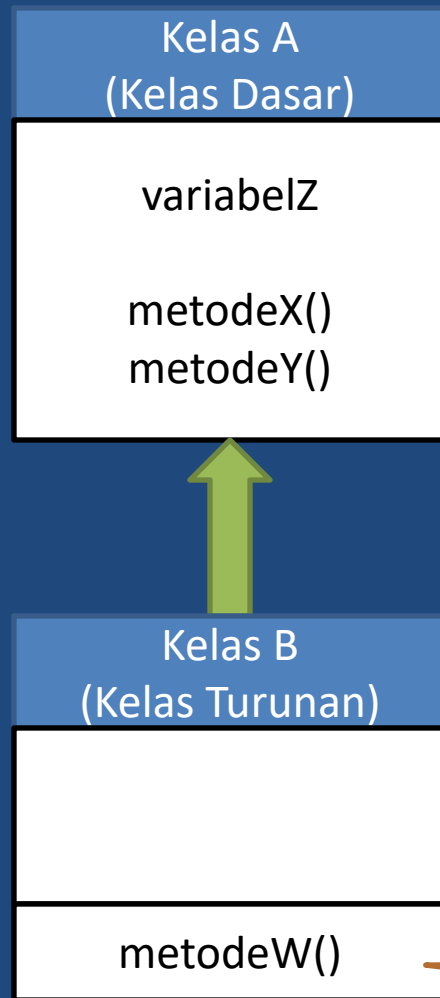

Inheritance



“Like Father Like Son”

```
Class childClass extends parentClass
```

Konsep Inheritance



Dengan sendirinya KelasB mewarisi semua metode dan variabel milik KelasA, **tentu saja yang tidak bersifat private.**

Metode tambahan

Inheritance

latihan5.php

```
2  class OrangTua {
3      public function helloOrangTua() {
4          echo "Ini adalah class OrangTua ...<br />";
5      }
6  }
7
8  class Anak extends OrangTua {
9      public function helloAnak() {
10         echo "Ini adalah class Anak ...<br />";
11     }
12 }
```

Inheritance

latihan5.php

```
14 $objekAnak = new Anak();  
15 $objekAnak->helloOrangTua();  
16 $objekAnak->helloAnak();
```

```
2 class OrangTua {
3     protected $nama= 'Leonardo';
4     public function helloOrangTua() {
5         echo "Ini adalah class OrangTua ...<br />";
6     }
7 }
8
9 class Anak extends OrangTua {
10     public function helloAnak() {
11         echo "Ini adalah class Anak ...<br />";
12     }
13     public function cetakNamaOrt() {
14         echo "Nama Orang Tua : $this->nama <br>";
15     }
16 }
```

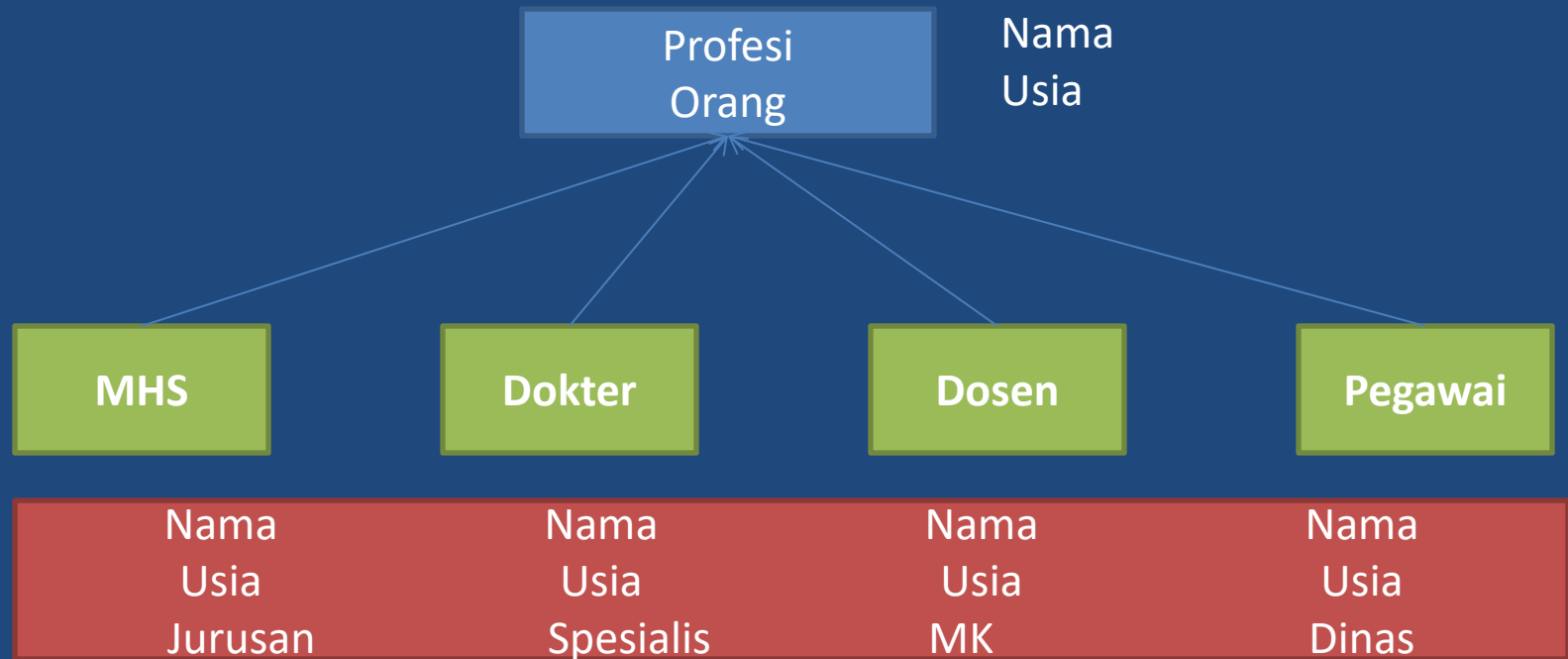
```
18 $objekAnak = new Anak();
19 $objekAnak->helloOrangTua();
20 $objekAnak->cetakNamaOrt();
21 $objekAnak->helloAnak();
```

Inheritance

latihan6.php

```
18 $objekAnak = new Anak();  
19 $objekAnak->helloOrangTua();  
20 $objekAnak->cetakNamaOrt();  
21 $objekAnak->helloAnak();
```

Contoh lain



Class in Module

Agar lebih efisien dalam pemrograman, kita bisa menaruh class didalam sebuah file tersendiri lalu meng-include file tersebut dalam setiap file PHP. Dengan demikian kita tidak perlu menulis ulang class dalam setiap file dan jika ada kesalahan atau modifikasi pada class, kita cukup merubah class dalam satu file tersebut.

```
<?php  
    include "class.php";  
  
    $object = new Class;  
?>
```


Autoload

Autoloading class adalah sebuah cara untuk memanggil sebuah class pada file lain tanpa menggunakan fungsi include

```
01. function __autoload($class_name) {  
02.     include $class_name . '.php';  
03. }
```

Autoload

Rumus1.php

```
01. <?php
02. class Rumus1 {
03.     var $panjang;
04.     var $lebar;
05.
06.     function luas() {
07.         return $this->panjang * $this->lebar;
08.     }
09.
10.     function __construct($x,$y) {
11.         $this->panjang = $x;
12.         $this->lebar = $y;
13.     }
14. }
15. ?>
```

Autoload

Rumus2.php

```
01. <?php
02. class Rumus2 {
03.     var $diameter;
04.
05.     function luas() {
06.         $r = $this->diameter / 2;
07.         return 3.14 * $r * $r;
08.     }
09.
10.     function __construct($x) {
11.         $this->diameter = $x;
12.     }
13. }
14. ?>
```

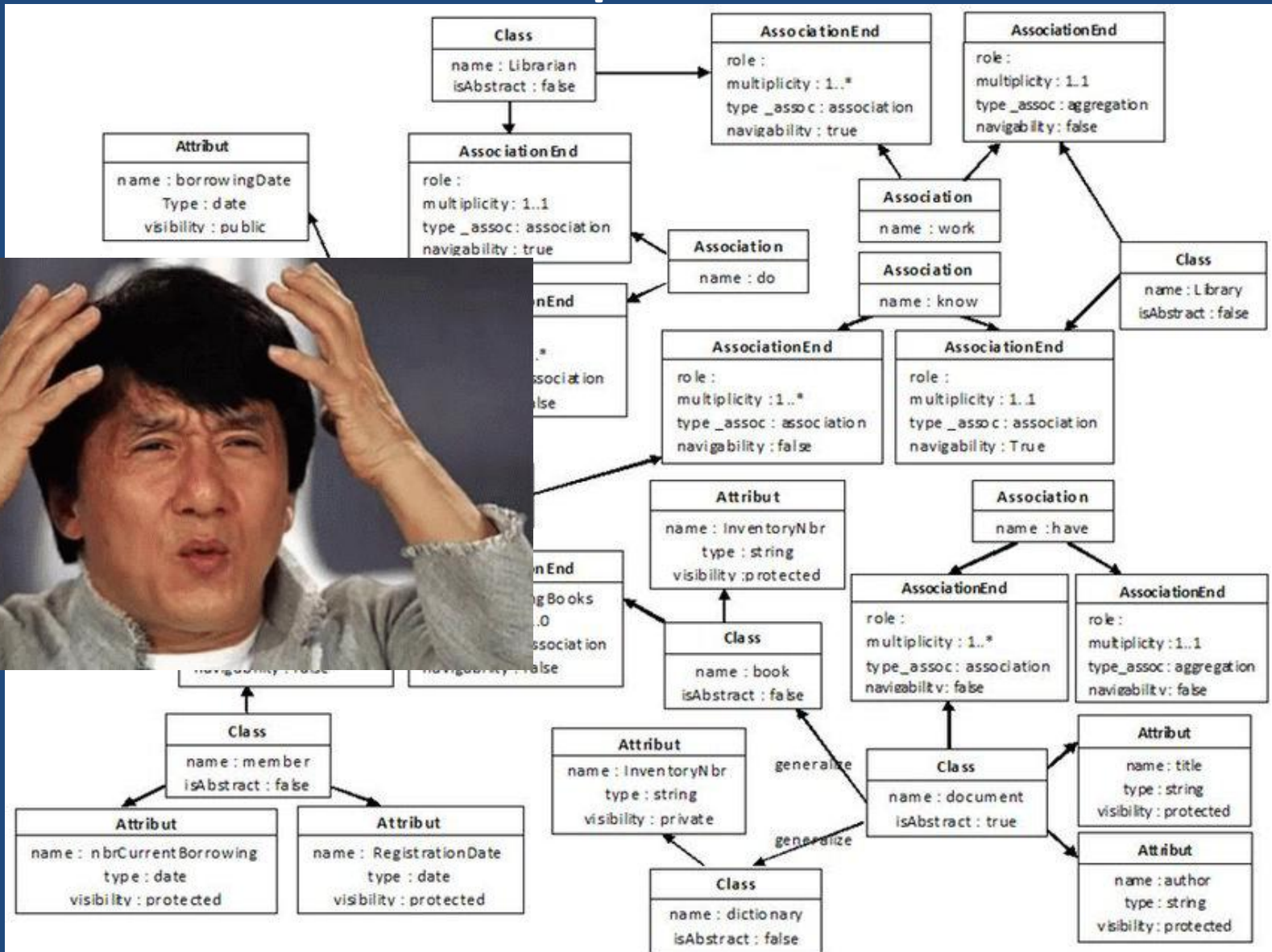
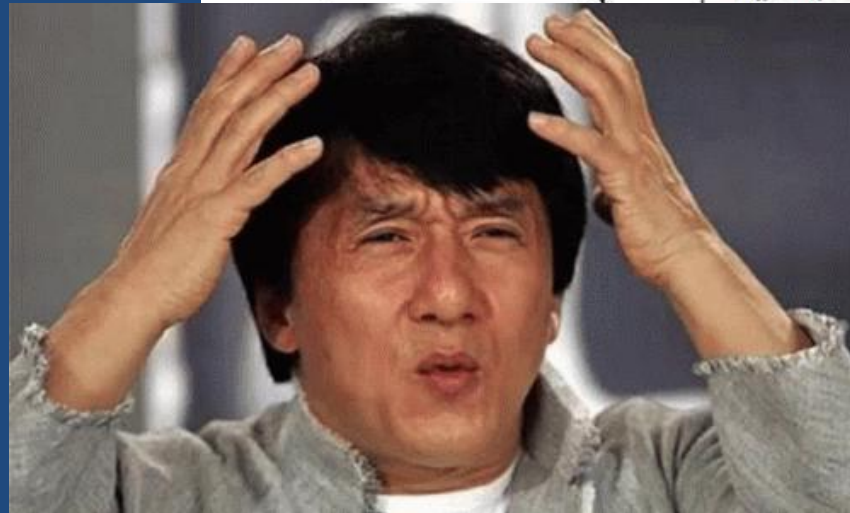
Autoload

Autoload.php

```
01. <?php
02. function __autoload($class_name) {
03.     include $class_name . '.php';
04. }
05.
06. $rumus1 = new Rumus1(5,8);
07. $rumus2 = new Rumus2(8);
08.
09. echo "Luas Rumus1 = ".$rumus1->luas()."<br/>";
10. echo "Luas Rumus2 = ".$rumus2->luas();
11. ?>
```

Pastikan ketiga file berada pada folder/direktori yang sama

Namespace



Namespace

GLOBAL NAMESPACE

RELATIVE NAMESPACE

Global Namespace

Class tanpa deklarasi namespace

namespacelab > 🐞 Angry.php

```
1  <?php
2
3  class Angry {
4
5      function __construct(){
6          echo "Summon Angry Mood\n";
7      }
8  }
```

```
1  <?php
2
3  include("angry.php");
4
5  $angry = new Angry();
```

Relative Namespace

```
1  <?php
2
3  namespace NiceMood;
4
5  class Happy {
6
7      function __construct(){
8          echo "Summon Happy Mood\n";
9      }
10 }
```

namespacelab > mood.php

```
1  <?php
2
3  include("Angry.php");
4  include("Happy.php");
5
6  $angry = new Angry();
7  $happy = new Happy();
```

PS D:\STSN\AJAR\WebPro\Lab\php\namespacelab> php mood.php

PHP Fatal error: Class 'Happy' not found in D:\STSN\AJAR\WebPro\Lab\php\namespacelab\mood.php on line 7

Fatal error: Class 'Happy' not found in D:\STSN\AJAR\WebPro\Lab\php\namespacelab\mood.php on line 7

Relative Namespace

```
1  <?php
2
3  namespace NiceMood;
4
5  class Happy {
6
7      function __construct(){
8          echo "Summon Happy Mood\n";
9      }
10 }
```

namespacelab > mood.php

```
1  <?php
2
3  include("Angry.php");
4  include("Happy.php");
5
6  $angry = new Angry();
7  $happy = new NiceMood\Happy();
```

Relative Namespace

namespacelab > Sad.php


```
1  <?php
2
3  namespace BadMood;
4
5  class Sad {
6
7      function __construct() {
8          echo "Summon Sad Mood\n";
9      }
10 }
```

namespacelab > Fair.php

```
1  <?php
2
3  namespace NiceMood;
4
5  include("Sad.php");
6  include("Happy.php");
7
8  class Fair {
9
10     function __construct(){
11         $sad = new Sad();
12         $happy = new Happy();
13     }
14 }
```

namespacelab > mood.php

```
1  <?php
2
3  include("Fair.php");
4
5  $fair = new NiceMood\Fair();
```

 \$sad = new \BadMood\Sad();

Namespace : Importing & Aliasing

```
namespacelab > mood.php
1  <?php
2
3  use NiceMood\Fair;
4  include("Fair.php");
5
6  $fair = new Fair();
```

Namespace : Importing & Aliasing

MyWeb\Blog\Post\Create;

MyWeb\Blog\Post\Update;

MyWeb\Blog\Post\Delete;

```
use MyWeb\Blog\Post as Post;
```

```
$create = new Post\Create();
```

```
$update = new Post\Update();
```

```
$delete = new Post\Delete();
```