Laporan Praktikum Pemrograman

Modul III –PENGENALAN AWAL PEMROGRAMAN BERORIENTASI OBJEK

(VISIBILITY, SETTERS, GETTERS, STATIC KEYWORD, DAN CONSTANT)

Raita Rahmatina/22103001006

Dosen : Achmad Arif Munaji, ST., M.Kom

Tanggal praktikum: 3 April 2023

Raita221201@gmail.com

Teknik Komputer

Institut Teknologi dan Sains Nahdlatul Ulama Kalimantan

***Abstrak*— Pada praktikum ketiga ini, membahas konsep dasar pemrograman berorientasi objek (OOP) dalam Bahasa PHP. Konsep yang dibahas mencakup tingkat visibilitas atau akses untuk properti dan metode kelas, *setters* dan *getters* untuk mengatur dan mengambil nilai properti, kata kunci statis untuk membuat properti atau metode terikat kelas, dan konstanta untuk membuat nilai tetap yang tidak dapat diubah saat *runtime*. Dalam pemrograman berorientasi objek, penting untuk memahami konsep dasar ini untuk membuat kode yang lebih efisien dan dapat dipelihara. Praktikum kali ini memberikan penjelasan lengkap dan contoh cara menerapkan setiap konsep, sehingga diharapkan mahasiswa memiliki pemahaman yang baik dan dapat menerapkan dalam kode yang dihasilkan.**

***Kata kunci—Visibility, Setter & Getters, Static Keyword, dan Constant***

1. Pendahuluan

*Object oriented programming* atau OOP adalah konsep pemrograman yang berorientasi pada objek atau data untuk mengatur sebuah desain program. Secara garis besar, cara kerja OOP yaitu membangun program dari gabungan objek-objek yang saling berhubungan dan disusun menjadi satu kesatuan. Bahasa pemrograman PHP sendiri sudah mendukung OOP sejak versi 5, sehingga programmer dapat menggunakan konsep OOP pada pembuatan aplikasi atau sistem dengan bahasa PHP. Konsep OOP pada bahasa PHP meliputi *visibility, setters, getters, static keyword,* dan *constant*.

*Visibility* adalah teknik yang digunakan untuk mengatur *property* maupun *method* pada sebuah objek. Dengan menggunakan *Access Modifier* nantinya akan memberi batasan akses yang di inginkan terhadap suatu objek tertentu. *Visibility* juga merupakan salah satu konsep yang digunakan untuk mengatur aksesibilitas suatu *property* atau *method* pada suatu *class*. Terdapat 3 jenis *visibility*, yaitu *public, private,* dan *protected*. *Setters* dan *getters* adalah metode yang digunakan untuk mengakses *property* pada suatu *class. Setters* digunakan untuk mengubah nilai *property* sedangkan *getters* digunakan untuk mengambil nilai property. *Static keyword* digunakan untuk membuat *property* atau *method* yang dapat diakses tanpa harus membuat *instance* dari *class* tersebut. Sedangkan *constant* adalah nilai yang tetap dan tidak dapat diubah pada suatu *class*.

Pada praktikum kali ini akan mempraktikan mengenai pengenalan awal tentang konsep-konsep dasar pada OOP pada bahasa pemrograman PHP. Khususnya mengenai *visibility, setters, getters, static keyword,* dan *constant*. Selain itu, kita juga akan mempelajari bagaimana cara menerapkan konsep-konsep tersebut pada bahasa PHP secara praktis.

1. Tinjauan Pustaka
   1. *Visibility*

*Visibility* adalah teknik yang digunakan untuk mengatur property maupun *method* pada sebuah objek. Dengan menggunakan *Access Modifier* nantinya akan memberi batasan akses yang di inginkan terhadap suatu objek tertentu. *Visibility* juga merupakan salah satu konsep yang digunakan untuk mengatur aksesibilitas suatu *property* atau *method* pada suatu class. Ada tiga tingkat *visibility* yang dapat diterapkan pada properti dan metode, yaitu:

*Public*,properti dan metode yang dideklarasikan sebagai *public* dapat diakses dari mana saja, baik dari dalam maupun luar *class*. *Protected*, properti dan metode yang dideklarasikan sebagai *protected* hanya dapat diakses oleh *class* itu sendiri atau oleh *class* turunan yang diturunkan dari *class* induknya. *Private*, properti dan metode yang dideklarasikan sebagai *private* hanya dapat diakses oleh *class* itu sendiri dan tidak dapat diakses dari luar *class*.

* 1. *Setters* dan *Getters*

*Setter* dan *getter* adalah metode dalam bahasa pemrograman PHP untuk mengakses dan mengubah nilai dari properti yang bersifat *private* atau *protected* pada suatu *class*. Dengan menggunakan *setter* dan *getter*, pengguna dapat membatasi akses langsung pada properti yang sensitif, sambil tetap memberikan kemudahan untuk mengubah dan membaca nilai dari properti tersebut.

*Setter* dan *getter* pada PHP biasanya didefinisikan dengan mengikuti konvensi penamaan yang sudah ditetapkan. Nama *setter* biasanya diawali dengan kata "*set*" dan diikuti dengan nama properti yang ingin diubah. Sedangkan, nama *getter* diawali dengan kata "*get*" dan diikuti dengan nama properti yang ingin dibaca.

* 1. *Static Keyword* dan *Constant*

*Static keyword* dan *constant* adalah fitur-fitur dalam bahasa pemrograman PHP yang digunakan untuk membuat properti dan metode yang bersifat konstan dan tetap dalam nilai atau perubahan. Keduanya memungkinkan pengguna untuk memperoleh nilai atau menggunakan metode tanpa harus membuat objek dari *class* yang bersangkutan.

*Static keyword* digunakan untuk membuat properti atau metode yang bersifat konstan pada *class*. Properti yang dideklarasikan sebagai *static* dapat diakses dari mana saja, tanpa harus membuat objek dari *class* tersebut. Demikian juga, metode yang dideklarasikan sebagai *static* dapat diakses tanpa harus membuat objek, sehingga memudahkan dalam penggunaan fungsionalitas tertentu tanpa perlu membuat *instance class* terlebih dahulu.

1. Langkah Praktikum

Pelaksanaan pada praktikum ini yaitu diawali dengan menjawab soal pretest yaitu mengenai *setters* dan *getters* digunakan untuk *visibility* apa? *Setters* dan *getters* pada PHP digunakan untuk mengakses dan mengubah nilai dari properti yang bersifat private atau *protected* pada suatu *class* dengan tetap menjaga *visibility*.

Kemudian, setelah soal *pretest* terjawab, mahasiswa membuka aplikasi *xampp control panel*, menyalakan di bagian *Apache* sampai warna hijau, membuka *browser* dengan mengetikkan *local host* dan membuat folder pada file *htdocs xampp* lalu membukanya dengan aplikasi Visual Stuido Code. Terakhir, menjawab soal praktikum :

1. Tambahkan *class Orange* yang merupakan *child class* dari *class Fruit*, kemudian tambahkan *property $taste* dan *method* *itsTaste()* yang menampilkan output dari *method get\_name()* dan *get\_color()* dengan menerapkan konsep *overriding*.
2. Buat agar *property $taste* dan *method itsTaste()* hanya bisa diakses oleh *class Orange* saja dan buat agar *property* dan *method* pada *class parent* hanya bisa diakses oleh *child* dan *parent* itu sendiri.
3. Buat *settters* dan *getters* untuk *property $taste* dan *method itsTaste()*.
4. Buat *property $height* yang berisikan nilai *integer* pada *class Orange* yang akan selalu tetap nilainya walaupun dilakukan *instance* berkali-kali.
5. Tugas 2:
   1. Tentukan kelas yang disebut Person.
   2. Karena semua Persons adalah dari species "Homo Sapiens", jadikan itu sebagai konstanta kelas
   3. Deklarasikan (tetapi belum ditentukan ) 3 properti kelas , dan \*Mereka semua harus \* . $name $age $occupationpublic
   4. Tentukan konstruktor kelas yang menerima tepat tiga argumen dengan urutan sebagai berikut: $name, $age, $occupationdan simpan di properti masing-masing.
   5. Tentukan metode yang dipanggil introduceyang tidak menerima argumen dan mengembalikan string format "Hello, my name is NAME\_HERE"
   6. Tetapkan metode lain yang disebut describe\_jobyang tidak menerima argumen dan mengembalikan format string "I am currently working as a(n) OCCUPATION\_HERE"( CATATAN: "a(n)"Bagian dari string adalah literal yang berarti Anda tidak perlu membuat persyaratan untuk memeriksa apakah "a"atau "an"harus digunakan. )
   7. Ketika makhluk luar angkasa tiba di Bumi, semua Persons diharapkan untuk menyambut mereka dengan cara yang persis sama. Oleh karena itu, tentukan metode kelas statis yang disebut greet\_extraterrestrials yang menerima argumen $species dan mengembalikan string format "Welcome to Planet Earth SPECIES\_NAME\_HERE!"
6. hasil dan analisis

Berikut penjelasan dan coding untuk soal 1 sampai 4 :

<?php

class Fruit {

  protected $name;

  protected $color;

  public function get\_name() {

    return $this->name;

  }

  public function get\_color() {

    return $this->color;

  }

}

class Orange extends Fruit {

  private static $height = 10;

  private $taste;

  public function \_\_construct($name, $color, $taste) {

    $this->name = $name;

    $this->color = $color;

    $this->taste = $taste;

  }

  public function itsTaste() {

    echo "This orange tastes ".$this->taste." and has a height of ".self::$height." cm.";

  }

  public function get\_taste() {

    return $this->taste;

  }

  public function set\_taste($taste) {

    $this->taste = $taste;

  }

}

$orange = new Orange("Orange", "Orange", "Sweet");

$orange->itsTaste();

echo "<br>";

echo $orange->get\_taste();

$orange->set\_taste("Sour");

echo "<br>";

echo $orange->get\_taste();

coding di atas dapat dijelaskan bahwa :

1. Pada soal pertama membuat *class Orange* (child class) dengan menambahkan *property* $taste dan *method* itsTaste(). Melakukan *overriding* pada *method* get\_name() dan get\_color() dari *parent class* (Fruit).
2. Pada soal kedua *Property* $taste dijadikan sebagai *private* dan *property* pada *parent class* diubah menjadi *protected* agar hanya bisa diakses oleh *child class*.
3. Pada soal ketiga *Setters* dan *getters* untuk *property* $taste dibuat agar nilai bisa diubah dan dibaca dengan cara yang terkontrol.
4. Dan pada soal keempat *Property* $height dijadikan sebagai *static* agar nilainya selalu tetap meskipun dilakukan *instance* berkali-kali.

Dibawah ini coding dan penjelasan untuk soal kelima (Tugas 2) :

<?php

class Person {

    const SPECIES = "Homo Sapiens";

    public $name;

    public $age;

    public $occupation;

    public function \_\_construct($name, $age, $occupation) {

      $this->name = $name;

      $this->age = $age;

      $this->occupation = $occupation;

    }

    public function introduce() {

      return "Hello, my name is " . $this->name;

    }

    public function describe\_job() {

      return "I am currently working as a(n) " . $this->occupation;

    }

    public static function greet\_extraterrestrials($species) {

      return "Welcome to Planet Earth " . $species . "!";

    }

  }

Baris 1-9: menjelaskan kelas *Person*, dengan properti $name, $age, dan $occupation yang semua bersifat publik.

Baris 2: menjelaskan konstanta kelas *Species* dengan nilai "*Homo Sapiens*".

Baris 11-16: menjelaskan konstruktor kelas yang menerima tiga argumen $name, $age, dan $occupation, dan menyimpannya di properti masing-masing.

Baris 18-21: menjelaskan metode *introduce* yang tidak menerima argumen dan mengembalikan string format "*Hello*, *my name is NAME\_HERE*" dengan menggabungkan properti $name.

Baris 23-26: menjelaskan metode *describe\_job* yang tidak menerima argumen dan mengembalikan *string format* "*I am currently working as a(n) OCCUPATION\_HERE*" dengan menggabungkan properti $occupation.

Baris 28-31: menjelaskan metode kelas statis *greet\_extraterrestrials* yang menerima satu argumen $species dan mengembalikan *string format* "*Welcome to Planet Earth SPECIES\_NAME\_HERE*!" dengan menggabungkan nilai $species.

1. Kesimpulan

Pemrograman berorientasi objek (OOP) pada bahasa pemrograman PHP melibatkan beberapa konsep dasar seperti visibility, setters, getters, static keyword, dan constant. Konsep-konsep tersebut penting dalam OOP karena dapat membantu dalam pengaturan nilai properti, aksesibilitas metode, serta pengelolaan nilai tetap yang tidak dapat diubah.

Praktikum selanjutnya diharapkan mahasaiswa dapat menyelesaikan penugasan secara lebih sederhana dan efektif dalam melaksanakan sistem pembelajaran yang ada.

VI. Daftar Pustaka

dewaweb.com

amuofficialnet.com

elib.unikom.ac.id

kodingin.com

php.net