Laporan Praktikum Pemrograman

Modul II – Pengenalan Awal Pemograman Berorientasi Objek (Constructor, Object Type, Inheritance, dan Overriding)

Muhammad Yusron Nashrulloh/22103001005

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

Tanggal praktikum: 27 Maret 2023

myusron434@gmail.com

Teknik Komputer

Institut Teknologi dan Sains Nahdlatul Ulama Kalimantan

***Abstrak*—Modul praktikum yang kedua ini membahas konsep dasar dari Pemrograman Berorientasi Objek dengan menggunakan bahasa pemrograman PHP. Praktikum ini bertujuan untuk memperkenalkan konsep-konsep dasar dalam pemrograman berorientasi objek, termasuk Constructor, Object Type, Inheritance, dan Overriding pada PHP. Dalam praktikum ini, mahasiswa akan belajar cara membuat constructor pada class, menginisialisasi objek dengan menggunakan constructor, serta mengimplementasikan inheritance atau pewarisan pada class untuk membuat class child dengan sifat dan perilaku yang sama dengan class parent. Selain itu, mahasiswa juga akan mempelajari bagaimana menimpa method yang ada pada parent class dengan method baru pada child class. Semua konsep tersebut akan diimplementasikan melalui praktikum dengan membuat class dan objek, serta melakukan pemrograman berorientasi objek dengan menggunakan bahasa pemrograman PHP. Setelah selesai menjalankan praktikum ini, mahasiswa diharapkan mampu memahami dan mengimplementasikan konsep dasar dari pemrograman berorientasi objek pada PHP.**

***Kata kunci— Constructor,Obejct Type, Inheritance & Overriding***

1. Pendahuluan

Pemrograman berorientasi objek (OOP) adalah paradigma pemrograman yang sangat populer dan penting dalam pengembangan perangkat lunak modern. Dalam OOP, konsep dasar yang harus dipahami meliputi objek, kelas, konstruktor, tipe objek, pewarisan (inheritance), dan penggantian (overriding). Konsep-konsep ini sangat penting untuk dipahami karena dapat membantu pengembang dalam membangun perangkat lunak yang efisien dan terstruktur dengan baik.

Pada modul praktikum kedua ini, akan membahas secara lebih mendalam konsep-konsep tersebut menggunakan bahasa pemrograman yang sangat populer yaitu PHP. Pertama-tama, akan membahas konstruktor, yaitu metode khusus dalam kelas yang digunakan untuk menginisialisasi objek. Selanjutnya, akan membahas tipe objek, yang merujuk pada jenis data yang dapat diatur oleh sebuah objek.

Kemudian, membahas pewarisan (inheritance), yang memungkinkan kelas baru untuk mengambil sifat dan metode dari kelas yang sudah ada. Dan terakhir, membahas penggantian (overriding), yang memungkinkan kelas turunan untuk mengganti metode yang sudah ada dalam kelas dasarnya.

1. Tinjauan Pustaka
   1. Constructor

Constructor pada PHP adalah metode khusus dalam sebuah kelas yang dipanggil secara otomatis ketika sebuah objek dibuat. Constructor digunakan untuk menginisialisasi nilai awal dari objek dan menetapkan nilai awal untuk properti objek.

* 1. Object Type

Object type pada PHP adalah tipe data yang merujuk pada objek tertentu dalam program. Object type menggambarkan jenis objek yang didefinisikan oleh kelas. Dalam PHP, object type digunakan untuk mengakses properti dan metode dari objek yang didefinisikan oleh kelas tertentu.

* 1. Inheritance dan Overriding

Inheritance pada PHP adalah konsep di mana sebuah kelas baru dapat mewarisi sifat dan metode dari kelas yang sudah ada. Dengan inheritance, kelas baru dapat memperluas atau mengubah perilaku dari kelas yang sudah ada tanpa harus menulis ulang kode dari awal. Dalam PHP, inheritance diimplementasikan dengan menggunakan kata kunci extends. Overriding pada PHP adalah konsep di mana sebuah kelas turunan dapat mengganti atau menimpa implementasi metode yang sudah ada pada kelas dasarnya. Dalam PHP, overriding dilakukan dengan mendefinisikan ulang metode yang sama pada kelas turunan.

1. Langkah Praktikum

Urutan pelaksanaan pada praktikum ini yaitu diawali dengan membuka aplikasi xampp control panel, lalu menyalakan di bagian Apache sampai warna hijau, setelah itu membuka browser dengan mengetikkan local host dan membuat folder pada file htdocs xampp lalu membukanya dengan aplikasi Visual Stuido Code. Kemudian mulai mengerjakan soal praktikum yaitu sebagai berikut :

1. Terapkan *constructor* pada *class Fruit*
2. Buat class baru bernama DetailFruit yang menerapkan konsep *Object Type class Fruit* pada *method* detail() dan kemudian *method* tersebut akan menampilkan output “I love $name and its $color”
3. Buat *Class Apple* dan *Class Banana* yang merupakan *Child* dari *Class Fruit*. Masing-masing *Child Class* mempunyai *property* $shape dan *method* detail() untuk menampilkan output: “I Love $name, its $color, and its $shape”. Terapkan juga konsep *Overriding* untuk membuat baris program menjadi lebih rapi dan tidak diulang-ulang.
4. Buat *Class* Kalkulator yang memiliki tiga *Child Class*, yaitu Tambah, Kurang, dan Bagi. *Parent Class* mempunyai *property* $angka1 dan $angka2. Semua *Class* mempunyai *method* hasil(). *Method* hasil() pada *Parent Class* hanya menampilkan angka saja tanpa dilakukan proses aritmatika, sedangkan *Child Class* melakukan proses aritmatika sesuai dengan nama *Class*-nya.
5. hasil dan analisis

Perintah atau soal yang dikerjakan pada praktikum modul 2 ini adalah sebagai berikut :

Terapkan *constructor* pada *class Fruit*

Buat class baru bernama DetailFruit yang menerapkan konsep *Object Type class Fruit* pada *method* detail() dan kemudian *method* tersebut akan menampilkan output “I love $name and its $color”

Buat *Class Apple* dan *Class Banana* yang merupakan *Child* dari *Class Fruit*. Masing-masing *Child Class* mempunyai *property* $shape dan *method* detail() untuk menampilkan output: “I Love $name, its $color, and its $shape”. Terapkan juga konsep *Overriding* untuk membuat baris program menjadi lebih rapi dan tidak diulang-ulang.

Buat *Class* Kalkulator yang memiliki tiga *Child Class*, yaitu Tambah, Kurang, dan Bagi. *Parent Class* mempunyai *property* $angka1 dan $angka2. Semua *Class* mempunyai *method* hasil(). *Method* hasil() pada *Parent Class* hanya menampilkan angka saja tanpa dilakukan proses aritmatika, sedangkan *Child Class* melakukan proses aritmatika sesuai dengan nama *Class*-nya.

Penjelasannya sebagai berikut :

<?php

//Tugas Nomor 1

?php

class Fruit {

    public $name;

    public $color;

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

      $this->name = $name;

      $this->color = $color;

    }

    function get\_name() {

      return $this->name;

    }

    function get\_color() {

      return $this->color;

    }

  }

  $apple = new Fruit("Apple", "Red");

  echo $apple->get\_name();

  echo "<br>";

  echo $apple->get\_color();

// Tugas Nomor2

class DetailFruit {

    public $fruit;

    function \_\_construct($fruit) {

      $this->fruit = $fruit;

    }

    function detail() {

      echo "I love " . $this->fruit->get\_name() . " and its " . $this->fruit->get\_color();

    }

  }

  $apple = new Fruit("Apple", "Red");

  $detail\_fruit = new DetailFruit($apple);

  $detail\_fruit->detail();

//Tugas Nomor 3

class Apple extends Fruit {

    public $shape;

    function \_\_construct($name, $color, $shape) {

      parent::\_\_construct($name, $color);

      $this->shape = $shape;

    }

    function detail() {

      echo "I love " . $this->name . ", its " . $this->color . ", and its " . $this->shape;

    }

  }

  class Banana extends Fruit {

    public $shape;

    function \_\_construct($name, $color, $shape) {

      parent::\_\_construct($name, $color);

      $this->shape = $shape;

    }

    function detail() {

      echo "I love " . $this->name . ", its " . $this->color . ", and its " . $this->shape;

    }

  }

  $apple = new Apple("Apple", "Red", "Round");

  $banana = new Banana("Banana", "Yellow", "Curved");

  $apple->detail();

  echo "<br>";

  $banana->detail();

//Tugas Nomor 4

<?php

class Kalkulator {

    protected $angka1;

    protected $angka2;

    public function \_\_construct($angka1, $angka2) {

        $this->angka1 = $angka1;

        $this->angka2 = $angka2;

    }

    public function hasil() {

        echo "{$this->angka1} dan {$this->angka2}";

    }

}

class Tambah extends Kalkulator {

    public function hasil() {

        echo $this->angka1 + $this->angka2;

    }

}

class Kurang extends Kalkulator {

    public function hasil() {

        echo $this->angka1 - $this->angka2;

    }

}

class Bagi extends Kalkulator {

    public function hasil() {

        echo $this->angka1 / $this->angka2;

    }

}

$kalkulator = new Kalkulator(10, 5);

$kalkulator->hasil(); // output: 10 dan 5

$tambah = new Tambah(10, 5);

$tambah->hasil(); // output: 15

$kurang = new Kurang(10, 5);

$kurang->hasil(); // output: 5

$bagi = new Bagi(10, 5);

$bagi->hasil(); // output: 2

Codingan diatas merupakan perintah yang harus dikerjakan pada praktikum ini, hasil dari codingan tersebut dapat dilihat pada chrome dengan hasil dari masing-masing codingan dari nomor tersebut penjelasannya sebagai berikut :

1. Pada perintah nomor 1 akan menghasilkan penerapan constructor pada class fruit yang berfungsi sebagai inisialisasi nilai awal dari objek dan menetapkan nilai awal untuk properti objek
2. Pada perintah nomor 2 membuat Detailfruit yang menerapkan konsep object type dan method akan menghasilkan output “I Love $Apple and its $red
3. Pada perintah nomor 3 masing-masing child class memiliki property shape dan method detail dengan menghasilkan output “I Love $Apple, its $Red and its $roundn dan “I Love $Banana, its $yellow and its $curved.

4) . Pada perintah nomor 4 Pada codingan di atas, terdapat pembuatan class Kalkulator sebagai parent class, dan tiga child class yaitu Tambah, Kurang, dan Bagi. Pada class Kalkulator terdapat dua property yaitu $angka1 dan $angka2, dan method \_\_construct() untuk menginisialisasi nilai property tersebut. Kemudian terdapat method hasil() pada parent class yang hanya menampilkan nilai property $angka1 dan $angka2 saja tanpa melakukan proses aritmatika.

Pada masing-masing child class, yaitu Tambah, Kurang, dan Bagi, terdapat method hasil() yang sesuai dengan nama class-nya. Ketika method hasil() dipanggil pada masing-masing child class, maka akan dilakukan proses aritmatika yang sesuai. Contoh penggunaannya adalah dengan membuat objek dari masing-masing child class, dan memanggil method hasil(). Sebagai contoh, dengan membuat objek $tambah dari class Tambah, dan memanggil method hasil(), maka akan dilakukan proses penjumlahan dari nilai property $angka1 dan $angka2 yang dimiliki oleh class Kalkulator.

1. Kesimpulan

Dari praktikum ini dapat disimpulkan bahwa pemrograman berorientasi objek (OOP) merupakan paradigma pemrograman yang sangat berguna untuk membuat program yang kompleks dan modular. Konsep dasar OOP, seperti Constructor, Object Type, Inheritance, dan Overriding pada PHP dapat membantu programmer dalam membuat kode yang lebih bersih, terstruktur, mudah dipahami.

VI. Daftar Pustaka

[***https://ejournal.undip.ac.id/index.php/js/article/view/3156/2979***](https://ejournal.undip.ac.id/index.php/js/article/view/3156/2979)

diakses pada tanggal 31 Maret 2023