|  |  |
| --- | --- |
|  | Jurusan Teknologi Informasi Politeknik Negeri Malang  **Jobsheet-12: PHP – OOP**  **Mata Kuliah Desain dan Pemrograman Web**  Pengampu: Tim Ajar Desain dan Pemrograman Web  *Oktober 2023* |

**Topik**

* Konsep Pembuatan Web Dinamis dengan OOP

**Tujuan**

Mahasiswa diharapkan dapat:

1. Mahasiswa mampu membuat class dan object, inheritance, polymorphism, encapsulation, abstraction, interfaces, constructors and destructors, dan encapsulation and access modifier
2. Mahasiswa mampu membuat CRUD dengan OOP

**Perhatian**

Jobsheet ini harus dikerjakan step-by-step sesuai langkah-langkah praktikum yang sudah diberikan. Soal dapat dijawab langsung di dalam kolom yang disediakan dengan menggunakan PDF Editor.

**Pendahuluan**

**OOP**

Pemrograman Berorientasi Objek (OOP) adalah paradigma pemrograman yang sangat penting dalam dunia pengembangan perangkat lunak. Ini memungkinkan para pengembang untuk mengorganisasi kode mereka menjadi objek-objek yang memiliki atribut (data) dan metode (fungsi) yang terkait.

**Pengenalan Pemrograman Berorientasi Objek (OOP)**

Pemrograman Berorientasi Objek didasarkan pada konsep objek, yang mewakili entitas dalam dunia nyata. Setiap objek memiliki karakteristik yang disebut atribut (properti), dan dapat melakukan tindakan tertentu yang disebut metode (fungsi). OOP membantu dalam memecah kode menjadi bagian-bagian yang lebih kecil dan lebih mudah dikelola.

## Kenapa OOP Penting?

Dalam dunia pengembangan website yang semakin kompleks dan dinamis, penggunaan Konsep Pemrograman Berorientasi Objek (OOP) telah menjadi landasan yang esensial. OOP membawa keefektifan, kemudahan pemeliharaan, dan skalabilitas yang tak ternilai harganya untuk proyek-proyek website. Artikel ini akan membahas mengapa OOP begitu penting dalam pengembangan proyek website dan manfaat utamanya.

### Modularitas dan Pengelolaan Kode yang Lebih Baik

Salah satu manfaat utama OOP adalah kemampuannya untuk memecah kode menjadi modul atau objek yang independen. Dalam pengembangan website, setiap komponen seperti formulir, tampilan, database, dan lainnya dapat diwakili sebagai objek yang terpisah. Ini memungkinkan tim pengembangan untuk bekerja secara terpisah pada komponen-komponen ini, mempercepat proses pengembangan dan memungkinkan pemeliharaan yang lebih mudah di masa depan.

### Penggunaan Ulang (Reusability) dan Efisiensi

Dalam OOP, objek-objek dapat digunakan ulang di berbagai bagian proyek. Ini mengurangi jumlah kode yang perlu ditulis, menghemat waktu dan usaha pengembang. Misalnya, jika Anda telah membuat objek "Formulir" yang memiliki metode untuk memvalidasi input, Anda dapat menggunakannya di berbagai halaman website tanpa perlu menulis ulang kode validasi tersebut.

### Pengelolaan Kesalahan yang Lebih Baik

Ketika terjadi kesalahan dalam kode OOP, Anda dapat dengan mudah mengisolasi dan menemukan sumber kesalahan tersebut karena setiap objek memiliki tanggung jawab yang jelas. Ini memungkinkan Anda untuk memperbaiki masalah lebih cepat dan lebih akurat, mengurangi waktu yang dihabiskan untuk debugging.

### Skalabilitas dan Pengembangan Kolaboratif

Proyek website cenderung berkembang seiring waktu. Dengan OOP, Anda dapat dengan mudah menambahkan fitur baru atau memperbarui komponen yang ada tanpa mengganggu fungsi lainnya. Tim pengembangan juga dapat bekerja secara paralel pada berbagai komponen, karena setiap objek berdiri sendiri dan tidak terlalu bergantung pada yang lain.

### Pemeliharaan Lebih Mudah

Ketika proyek website tumbuh, pemeliharaan menjadi sangat penting. OOP membantu dalam memisahkan perubahan yang diperlukan pada suatu komponen tanpa mempengaruhi yang lain. Jika Anda ingin mengubah tampilan halaman tertentu, Anda hanya perlu mengedit objek tampilan tanpa perlu khawatir tentang dampaknya pada komponen lain.

### Enkapsulasi dan Keamanan

Konsep enkapsulasi dalam OOP memungkinkan Anda untuk menyembunyikan detail implementasi dari komponen lainnya. Ini berarti bahwa komponen lain hanya dapat berinteraksi dengan objek melalui antarmuka yang ditentukan, mengurangi potensi kesalahan atau manipulasi yang tidak diinginkan.

### Fleksibilitas dan Peningkatan Kualitas Kode

OOP memungkinkan Anda untuk membuat abstraksi yang tinggi untuk mengelola kerumitan dan mendefinisikan pola umum. Ini meningkatkan kualitas kode karena mengikuti prinsip-prinsip yang terbukti dalam desain perangkat lunak, seperti DRY (Don't Repeat Yourself) dan SOLID (Prinsip-responsibilitas terpisah, Terbuka-Tertutup, Substitusi Liskov, Segregasi Antarmuka, Ketergantungan Inversi).

## Konsep Utama dalam OOP PHP

Dalam PHP, OOP memungkinkan Anda untuk mengorganisir dan mengelompokkan kode menjadi unit-unit yang lebih terstruktur dan mudah dikelola. Berikut adalah konsep-konsep utama OOP dalam PHP:

**Praktikum 1. Basic OOP**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Kelas adalah blueprint atau cetak biru yang mendefinisikan struktur dan perilaku suatu objek. Kelas berisi atribut (data) dan metode (fungsi) yang berkaitan dengan objek tersebut. Objek, di sisi lain, adalah instance konkret dari suatu kelas, memiliki nilai nyata untuk atribut dan mampu menjalankan metode yang didefinisikan dalam kelas. Dalam PHP, Anda dapat membuat kelas dengan kata kunci class dan kemudian membuat objek dari kelas tersebut dengan kata kunci new. Berikut adalah contoh sederhana: |
| 2 | Buatlah folder oop dalam folder dasarWeb/ dengan file baru yaitu oop.php. |
| 3 | Ketikkan ke dalam file oop.php tersebut kode di bawah ini. |
| 4 |  |
| 5 | Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.1)  Jawab: program PHP ini mendefinisikan sebuah **kelas** bernama Car yang berfungsi sebagai **blueprint** untuk membuat **objek** mobil.   * **Kelas**Car**:** Merupakan template yang mendefinisikan properti dan perilaku yang dimiliki oleh semua objek mobil. * **Properti**brand**:** Merupakan variabel publik yang menyimpan nilai merek mobil (misalnya "Toyota" atau "Honda"). * **Metode**startEngine()**:** Merupakan fungsi publik yang menampilkan pesan "Engine started!" untuk mensimulasikan mesin mobil yang hidup.   Program ini kemudian membuat dua **instance** (objek) dari kelas Car: $car1 dan $car2. Setiap instance memiliki properti brand yang bisa diset secara independen. Terakhir, program memanggil metode startEngine() pada $car1 dan menampilkan nilai properti brand dari $car2, yang menunjukkan kemampuan untuk mengakses informasi dari instance yang berbeda. |
| 6 | Inheritance adalah salah satu konsep dasar dalam pemrograman berorientasi objek (OOP) yang memungkinkan sebuah class untuk mewarisi properti dan metode dari class lain. Class yang mewarisi disebut subclass atau child class, sedangkan class yang memberikan warisan disebut superclass atau parent class. Konsep ini memungkinkan kita untuk menggunakan kembali kode, memperpanjang fungsionalitas, dan membangun hierarki class.  Berikut ini adalah contoh sederhana penerapan inheritance dalam PHP:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.2)  Jawab: Baik, program PHP ini menunjukkan bagaimana konsep pewarisan (inheritance) digunakan untuk membuat hirarki kelas yang mewakili hewan.   * **Kelas Basis (Animal):**   + Merupakan kelas dasar yang menyimpan properti dan metode umum untuk semua hewan.   + Variabel terproteksi $name menyimpan nama hewan, bisa diakses oleh kelas ini dan kelas turunannya.   + Konstruktor publik \_\_construct(nama) menginisialisasi properti $name saat objek dibuat.   + Metode publik:     - eat(): Mencetak pesan yang menunjukkan hewan sedang makan.     - sleep(): Mencetak pesan yang menunjukkan hewan sedang tidur. * **Kelas Turunan (Cat dan Dog):**   + Menerima properti dan metode dari kelas Animal melalui pewarisan, sehingga kode dapat dipakai ulang.   + Cat: Merupakan kelas turunan dari Animal dengan metode khusus meow() untuk mensimulasikan suara meow kucing.   + Dog: Merupakan kelas turunan dari Animal dengan metode khusus bark() untuk mensimulasikan suara gonggongan anjing. * **Pembuatan Objek dan Pemanggilan Metode:**   + Program membuat instance (objek) dari kelas Cat dan Dog, dinamai $cat dan $dog.   + Kedua objek dapat menggunakan metode yang diwarisi dari Animal, sehingga mereka bisa melakukan tindakan umum seperti makan dan tidur menggunakan $cat->eat() dan $dog->sleep().   + Selain itu, $cat dapat menjalankan metode uniknya meow(), sementara $dog dapat memanggil metode spesifiknya bark(), menunjukkan kekuatan pewarisan untuk memodelkan perilaku berbeda pada kelas khusus. |
| 6 | Polymorphism adalah konsep dalam pemrograman berorientasi objek yang memungkinkan objek dari class yang berbeda untuk merespon pada pemanggilan metode dengan cara yang sama. Ini dapat diwujudkan dalam PHP melalui penggunaan antarmuka (interface) dan penggunaan overriding metode. Dengan polymorphism, Anda dapat memperlakukan objek dari class yang berbeda dengan cara yang seragam.  Berikut adalah contoh sederhana penggunaan polymorphism dalam PHP menggunakan antarmuka:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.3)  Jawab:  Program PHP ini mendemonstrasikan konsep **interface** untuk menerapkan konsep **abstraksi** dan **polimorfisme** dalam perhitungan luas bangun datar.   * **Interface** Shape**:**   + Merupakan kontrak yang mendefinisikan fungsionalitas wajib yang harus dimiliki oleh semua kelas bentuk (shape).   + Ia memiliki satu metode publik calculateArea(), yang mengharuskan setiap kelas yang mengimplementasikan interface ini untuk menghitung luasnya sendiri. * **Kelas** Circle **dan** Rectangle**:**   + Merupakan kelas konkrit yang mengimplementasikan interface Shape.   + Keduanya memiliki properti pribadi untuk menyimpan informasi spesifik:     - Circle: $radius untuk menyimpan nilai radius lingkaran.     - Rectangle: $width dan $height untuk menyimpan lebar dan tinggi persegi panjang.   + Kedua kelas mengimplementasikan metode calculateArea() sesuai dengan rumus luas masing-masing bentuk:     - Circle: Menggunakan pi() dan pow() untuk menghitung luas lingkaran (π \* radius^2).     - Rectangle: Menggunakan perkalian untuk menghitung luas persegi panjang (lebar \* tinggi). * **Fungsi** printArea**:**   + Merupakan fungsi yang menerima parameter bertipe Shape.   + Fungsi ini bersifat polimorfis, artinya dapat bekerja dengan objek dari berbagai kelas selama mereka mengimplementasikan interface Shape dan memiliki metode calculateArea().   + Fungsi ini memanggil calculateArea() pada objek yang diterimanya dan mencetak hasil perhitungan luasnya. * **Pembuatan Objek dan Penggunaan Fungsi:**   + Program membuat objek $circle dari kelas Circle dan objek $rectangle dari kelas Rectangle dengan nilai yang sesuai.   + Fungsi printArea dipanggil dua kali, sekali dengan parameter $circle dan sekali dengan parameter $rectangle.   + Karena keduanya mengimplementasikan interface Shape dan memiliki metode calculateArea(), fungsi printArea dapat bekerja dengan kedua objek tersebut dan menampilkan luas masing-masing bentuk. |
| 7 | Encapsulation adalah salah satu konsep dalam pemrograman berorientasi objek (OOP) yang mengizinkan pembungkusan (encapsulation) properti dan metode dalam sebuah class sehingga akses ke mereka dapat dikontrol. Hal ini dapat membantu dalam menerapkan prinsip-prinsip pengelolaan akses dan memastikan bahwa properti dan metode yang mungkin berubah di kemudian hari tidak merusak integritas class atau program secara keseluruhan.  Berikut adalah contoh sederhana encapsulation dalam PHP:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.4)  Jawab: Program ini mencontohkan penggunaan kelas untuk merangkum data dan fungsionalitas terkait objek. Kelas Car menyimpan informasi model dan warna mobil, serta menyediakan metode untuk mengakses dan mengubahnya. Ini membantu menjaga kode terorganisir dan mudah dikelola, terutama jika Anda ingin membuat banyak objek mobil dengan informasi berbeda. |
| 8 | Abstraction adalah salah satu konsep dasar dalam pemrograman berorientasi objek (OOP) yang memungkinkan Anda menyembunyikan detail internal dan hanya mengekspos fungsionalitas yang diperlukan. Ini membantu dalam menciptakan class dan metode yang bersifat umum dan fleksibel, memungkinkan pengguna untuk berinteraksi dengan objek tanpa perlu mengetahui implementasi internalnya.  Berikut adalah contoh sederhana abstraksi dalam PHP menggunakan abstract class dan method:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.5)  Jawab: Kelas abstrak berguna untuk menentukan kontrak yang harus diikuti oleh subkelas. Subkelas konkrit mewarisi fungsionalitas dari kelas abstrak dan menyediakan implementasi untuk metode abstrak. Pendekatan ini meningkatkan penggunaan kembali kode dan memastikan konsistensi dalam cara bentuk menangani perhitungan luas. |
| 9 | Interface adalah konsep dalam pemrograman berorientasi objek yang memungkinkan definisi kontrak atau kerangka yang harus diikuti oleh class-class yang mengimplementasikannya. Interface tidak memiliki implementasi sendiri, tetapi hanya menyediakan deklarasi metode dan properti yang harus diimplementasikan oleh class yang menggunakannya. Hal ini memungkinkan untuk mencapai polimorfisme tanpa memerlukan pewarisan tunggal, sehingga sebuah class dapat mengimplementasikan beberapa interface.  Berikut adalah contoh penggunaan interface dalam PHP:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.6)   * Jawab: Interface berguna untuk membuat kontrak, memastikan perilaku yang konsisten di seluruh kelas yang mengimplementasikannya. * Polimorfisme memungkinkan kode untuk bekerja dengan objek dari tipe berbeda selama mereka berbagi interface yang sama. Ini meningkatkan fleksibilitas dan penggunaan kembali kode. |
| 10 | Constructors dan destructors adalah metode khusus dalam pemrograman berorientasi objek (OOP) yang digunakan dalam PHP untuk menginisialisasi dan membersihkan objek. Constructor adalah metode yang dipanggil secara otomatis ketika objek baru dibuat, sedangkan destructor adalah metode yang dipanggil secara otomatis ketika objek dihapus atau tidak lagi digunakan.  Constructor (Metode Pembuat)  Constructor menggunakan nama khusus \_\_construct dalam PHP. Constructor ini akan dipanggil secara otomatis setiap kali objek baru dibuat dari class yang mengandung constructor tersebut.  Destructor (Metode Penghancur)  Destructor menggunakan nama khusus \_\_destruct dalam PHP. Destructor ini akan dipanggil secara otomatis ketika objek dihapus atau program selesai dieksekusi.  Berikut adalah contoh constructor dan destructor:    Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.7)  Jawab: Kelas sebagai Blueprint: Kelas mendefinisikan properti dan metode yang akan dimiliki objek dari tipe tersebut.  Konstruktor: Metode khusus yang digunakan untuk menginisialisasi objek selama pembuatan.  Metode Publik: Metode yang dapat diakses dari luar kelas untuk berinteraksi dengan properti objek.  Destruktor: Metode khusus yang dapat digunakan untuk melakukan pembersihan saat objek dihancurkan. |
| 11 | Encapsulation and Access Modifiers Encapsulation adalah salah satu konsep utama dalam pemrograman berorientasi objek (OOP), dan itu melibatkan pembungkusan data (variabel) dan metode (fungsi) dalam sebuah class. Ini membantu dalam menyembunyikan implementasi internal suatu class dan hanya mengekspos fungsionalitas yang diperlukan. Access modifiers adalah bagian dari encapsulation yang memungkinkan Anda mengontrol tingkat akses ke properti dan metode dalam sebuah class.  PHP memiliki tiga access modifiers utama yang dapat digunakan dalam class:  Public (public): Properti atau metode yang dideklarasikan sebagai public dapat diakses dari luar class, sehingga mereka bersifat terbuka untuk diakses dari mana saja.  Protected (protected): Properti atau metode yang dideklarasikan sebagai protected hanya dapat diakses dari dalam class itu sendiri dan dari class turunannya (inheritance).  Private (private): Properti atau metode yang dideklarasikan sebagai private hanya dapat diakses dari dalam class itu sendiri. Mereka tidak dapat diakses dari luar class, bahkan oleh class turunannya.  Berikut adalah contoh penggunaan access modifiers dalam PHP:  Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 1.8)  Jawab: Program PHP ini menunjukkan konsep tingkat akses properti (publik, dilindungi, privat) dalam sebuah kelas bernama Animal2. Ini membuat objek yang mewakili hewan dengan properti untuk nama, umur, dan warna. |

**Praktikum 2. CRUD dengan OOP**

|  |  |
| --- | --- |
| **Langkah** | **Keterangan** |
| 1 | Buat file baru pada folder oop dengan nama baru bernama database.php. Ketikkan kode seperti di bawah ini. |
| 2 |  |
| 3 | Buat file baru pada folder oop dengan nama baru bernama crud.php. Ketikkan kode seperti di bawah ini. |
| 4 |  |
| 5 | Buat file baru pada folder oop dengan nama baru bernama index.php. Ketikkan kode seperti di bawah ini. |
| 6 |  |
| 7 | Buat file baru pada folder oop dengan nama baru bernama edit.php. Ketikkan kode seperti di bawah ini. |
| 8 | Jalankan code pada praktikum 2. Apa yang anda pahami dari code di atas. Catat di bawah ini pemahaman anda. (soal no 2.1)  Jawab: Program tersebut merupakan program yang menggunakan CRUD dengan OOP |