Laporan Praktikum

Mata Kuliah Pemrograman Berorientasi Objek

Pertemuan 5

POLYMORPHISM

Dosen Pengampu:
Wildan Aprizal Arifin, S.Pd., M.Kom

Disusun Oleh:
Nelita Maharani
2307052

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

JavaScript, sebagai bahasa pemrograman yang serbaguna, menawarkan dukungan yang kuat untuk pemrograman berorientasi objek (OOP) melalui fitur-fitur utamanya. Dengan menggunakan 'class' sebagai cetak biru untuk menciptakan objek, pengembang dapat mendefinisikan properti dan method yang akan dimiliki oleh instance objek tersebut. Konsep pewarisan (inheritance) memungkinkan sebuah class untuk mewarisi karakteristik dari class lain, menciptakan hierarki yang memungkinkan penggunaan ulang kode secara efisien. Sementara itu, polimorfisme memberikan fleksibilitas dengan memungkinkan class-class turunan untuk menyediakan implementasi yang berbeda untuk method yang sama, memungkinkan objek-objek yang berbeda merespons dengan cara yang unik terhadap pemanggilan method yang identik. Ketiga konsep ini - class, pewarisan, dan polimorfisme - bekerja bersama untuk memungkinkan pengembang menciptakan kode yang lebih terorganisir, dapat digunakan kembali, dan mudah dipelihara, menjadikan JavaScript pilihan yang powerful untuk pengembangan aplikasi berbasis objek.

II. ALAT DAN BAHAN

1. Komputer/Laptop
2. Visual Studio Code
3. Git Hub
4. W3School
   Polymorphism

III. LANGKAH KERJA

1. Membuat class dasar.
2. Mengimplementasikan inheritance/warisan.
3. Menerapkan polymorphism.
4. Menggunakan getter dan setter.
5. Tambahkan statis metode.

IV. KESIMPULAN

Object-Oriented Programming (OOP) dalam JavaScript telah terbukti menjadi paradigma yang sangat berharga dalam pengembangan aplikasi modern. Melalui penerapan tiga konsep fundamental - Class sebagai cetak biru untuk objek, Inheritance yang memungkinkan pewarisan sifat antar class, dan Polymorphism yang memberikan fleksibilitas dalam implementasi method - JavaScript menawarkan cara yang efektif untuk mengorganisir dan menstrukturkan kode. Pendekatan ini menghasilkan sejumlah keuntungan signifikan, termasuk modularitas yang meningkatkan kemudahan dalam pengelolaan kode, reusabilitas yang memungkinkan penggunaan kembali komponen di berbagai bagian aplikasi, serta skalabilitas dan maintainability yang memudahkan pengembangan dan pemeliharaan aplikasi berskala besar. Best practices seperti penggunaan constructor untuk inisialisasi properti, pemanfaatan getter/setter untuk kontrol akses yang lebih baik, dan penerapan prinsip Single Responsibility, telah menjadi standar dalam pengembangan berbasis OOP. Meskipun demikian, penting untuk diingat bahwa OOP bukanlah satu-satunya paradigma dalam JavaScript, dan penggunaannya harus disesuaikan dengan kebutuhan proyek. Untuk pengembangan lebih lanjut, pemahaman tentang design patterns, penerapan testing untuk code berbasis OOP, dan praktik langsung dalam proyek nyata akan sangat bermanfaat. Secara keseluruhan, OOP dalam JavaScript memberikan fondasi yang kokoh untuk membangun aplikasi yang kompleks namun terstruktur, memungkinkan pengembang untuk menciptakan kode yang tidak hanya efisien dan dapat digunakan kembali, tetapi juga mudah dipelihara dan dikembangkan seiring waktu.

Link github subclass, inheritance, dan polymorphism:

https://github.com/maharaninelita/PBO_subclass_methode.git