Laporan Praktikum

Mata Kuliah Pemograman Berorientasi Objek

Pertemuan 5 Tugas ke 4

“polymorphysm”

Dosen Pengampu :

Willdan Aprizal Arifin, S.Pd., M.Kom

Disusun Oleh :

Najwa Lutfi Pramesthi (2300923)

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

Tujuan

Memahami dan mengimplementasikan konsep **polymorphism** menggunakan **class** pada JavaScript dengan membuat superclass dan subclass yang menerapkan metode yang sama namun menghasilkan output berbeda.

Langkah-Langkah Percobaan

Berikut adalah penjelasan yang ada:

1. Kelas Kapal (Induk)
Kelas `Kapal` merupakan dasar dari semua jenis kapal. Kelas ini memiliki properti:
- `nama`: Nama kapal
- `jenis`: Jenis kapal
- `panjang`: Panjang kapal (dalam meter)
- `lebar`: Lebar kapal (dalam meter)

Kelas ini juga memiliki metode `infoKapal()` yang mengembalikan string dengan informasi dasar tentang kapal.

2. Kelas Turunan
Kelas turunan dari `Kapal` adalah spesialisasi dari kelas induk, dengan tambahan properti yang lebih spesifik sesuai dengan jenis kapal tersebut.

-KapalFerry: Mewakili kapal ferry, dengan tambahan properti `kapasitas`, yang menunjukkan berapa banyak penumpang yang bisa diangkut.
-KapalPesiar: Mewakili kapal pesiar, dengan tambahan properti `jumlahKabin`, yang menunjukkan berapa banyak kabin yang ada di kapal.
-KapalTunda: Mewakili kapal tunda, yang memiliki tambahan properti `dayaTunda`, yaitu seberapa besar daya tunda kapal dalam satuan ton.
-KapalTanker: Mewakili kapal tanker, dengan properti tambahan `kapasitasTanki`, yaitu kapasitas tanki kapal dalam liter untuk mengangkut cairan (misalnya minyak).
Setiap kelas turunan meng-overwrite (menimpa) metode `infoKapal()` dari kelas induk, menambahkan informasi spesifik kapal tersebut.

3. Penggunaan Objek
Setelah kelas-kelas tersebut dibuat, objek-objek individual dari masing-masing kapal dibuat, misalnya:
- `kapalFerry` yang merepresentasikan kapal ferry dengan nama "Budiyono", kapasitas 600 orang, panjang 200 meter, dan lebar 100 meter.
- `kapalPesiar` yang merepresentasikan kapal pesiar "Titanic" dengan 1000 kabin.
- `kapalTunda` dengan daya tunda 5000 ton.
- `kapalTanker` dengan kapasitas tanki sebesar 500.000 liter.

4. Output
Setelah objek dibuat, program menampilkan informasi dari setiap kapal menggunakan metode `infoKapal()`. Sebagai contoh, output dari `kapalFerry.infoKapal()` akan mengembalikan kalimat seperti :
Kapal Budiyono merupakan jenis Ferry yang berukuran 200 M x 100m. Kapal ini memiliki kapasitas 600 orang.

Hal ini menunjukkan bahwa setiap objek dapat menampilkan informasi lengkap tentang jenis dan spesifikasi kapal tersebut.

```
class Kapal {
    constructor(nama, jenis, panjang, lebar) {
      this.nama = nama;
      this.jenis = jenis;
      this.panjang = panjang;
      this.lebar = lebar;
    }
    infoKapal() {
      return `Kapal ${this.nama} merupakan jenis ${this.jenis} yang berukuran ${this.panjang} M x
${this.lebar}m.`;
    }
  }
  class KapalFerry extends Kapal {
    constructor(nama, jenis, panjang, lebar, kapasitas) {
      super(nama, jenis, panjang, lebar);
      this.kapasitas = kapasitas;
    }

    infoKapal() {
      return `${super.infoKapal()} Kapal ini memiliki kapasitas ${this.kapasitas} orang.`;
    }
  }

  class KapalPesiar extends Kapal {
    constructor(nama, jenis, panjang, lebar, jumlahKabin) {
      super(nama, jenis, panjang, lebar);
      this.jumlahKabin = jumlahKabin;
    }
    infoKapal() {
      return `${super.infoKapal()} Kapal ini memiliki ${this.jumlahKabin} kabin.`;
    }
  }
  class KapalTunda extends Kapal {
    constructor(nama, jenis, panjang, lebar, dayaTunda) {
      super(nama, jenis, panjang, lebar);
      this.dayaTunda = dayaTunda;
    }
    infoKapal() {
      return `${super.infoKapal()} Kapal ini memiliki daya tunda ${this.dayaTunda} ton.`;
    }
  }
  class KapalTanker extends Kapal {
    constructor(nama, jenis, panjang, lebar, kapasitasTanki) {
      super(nama, jenis, panjang, lebar);
      this.kapasitasTanki = kapasitasTanki;
    }
    infoKapal() {
      return `${super.infoKapal()} Kapal ini memiliki kapasitas tanki ${this.kapasitasTanki} liter.`;
    }
  }

  // Buat objek dari setiap kelas
  const kapalFerry = new KapalFerry("Budiyono", "Ferry", 200, 100, 600);
  const kapalPesiar = new KapalPesiar("Titanic", "Kapal Pesiar", 300, 150, 1000);
  const kapalTunda = new KapalTunda("Tunda 1", "Kapal Tunda", 50, 20, 5000);
  const kapalTanker = new KapalTanker("Tanker 1", "Tanker", 250, 120, 500000);

  // Tampilkan informasi tentang setiap kapal
  console.log(kapalFerry.infoKapal());
  console.log(kapalPesiar.infoKapal());
```

```
console.log(kapalTunda.infoKapal());
console.log(kapalTanker.infoKapal());
```

Program di atas menggunakan konsep inheritance (pewarisan) pada pemrograman berorientasi objek dengan bahasa JavaScript. Pada dasarnya, terdapat sebuah kelas induk bernama `Kapal`, yang kemudian diikuti oleh beberapa kelas turunan seperti `KapalFerry`, `KapalPesiar`, `KapalTunda`, dan `KapalTanker`. Setiap kelas turunan mewarisi properti dan metode dari kelas induknya, namun menambahkan fitur spesifik masing-masing.

Kesimpulan

-Pewarisan: Kelas turunan (`KapalFerry`, `KapalPesiar`, dll.) mewarisi properti dan metode dari kelas induk `Kapal`, namun menambahkan detail sesuai jenisnya.

-Polimorfisme: Metode `infoKapal()` di setiap kelas turunan ditulis ulang (overridden) agar menampilkan informasi tambahan yang lebih spesifik sesuai dengan jenis kapal.

- Pembuatan Objek: Dari setiap kelas, kita bisa membuat objek individual dengan spesifikasinya masing-masing.