




# **RELASI ANTAR KELAS**

Nugroho D.S.


---



Konsep pemrograman berorientasi objek mengambil realita dari dunia nyata. Setiap objek di dunia nyata memiliki hubungan dengan objek lain, baik hubungan yang bersifat kuat maupun yang bersifat lemah. Begitu pula dengan sebuah aplikasi dibangun dengan menggabungkan beberapa kelas. Kelas-kelas tersebut saling bekerjasama untuk menyelesaikan suatu masalah. Dalam aplikasi yang berukuran yang cukup kompleks, banyak kelas-kelas yang terlibat dalam aplikasi tersebut.

# Pendahuluan

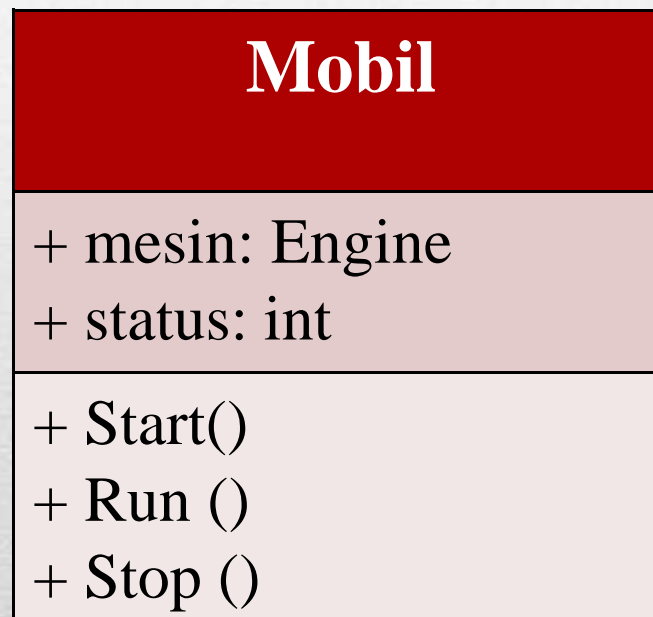
---

- 
- Pemodelan digunakan untuk menggambarkan desain sistem.
  - UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram.
  - UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

# Pemodelan UML

---

- Kelas dalam UML dimodelkan dalam bentuk persegi yang terdiri dari 3 bagian yaitu Nama Kelas, properti dan method yang dimiliki oleh kelas tersebut.



# Pemodelan UML

---





Terdapat beberapa macam relasi antar kelas yaitu:

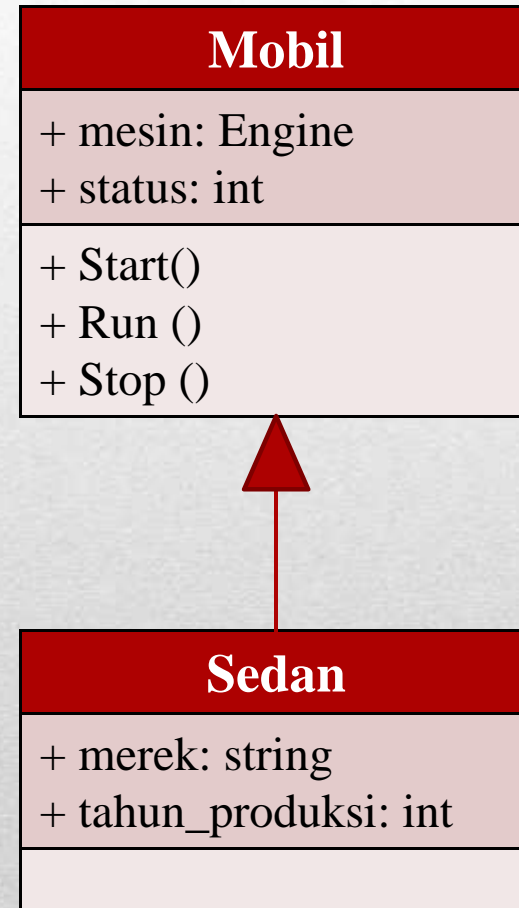
- Inheritance
- Realization
- Dependency
- Aggregation
- Composition

# **Relasi antar Kelas**

---

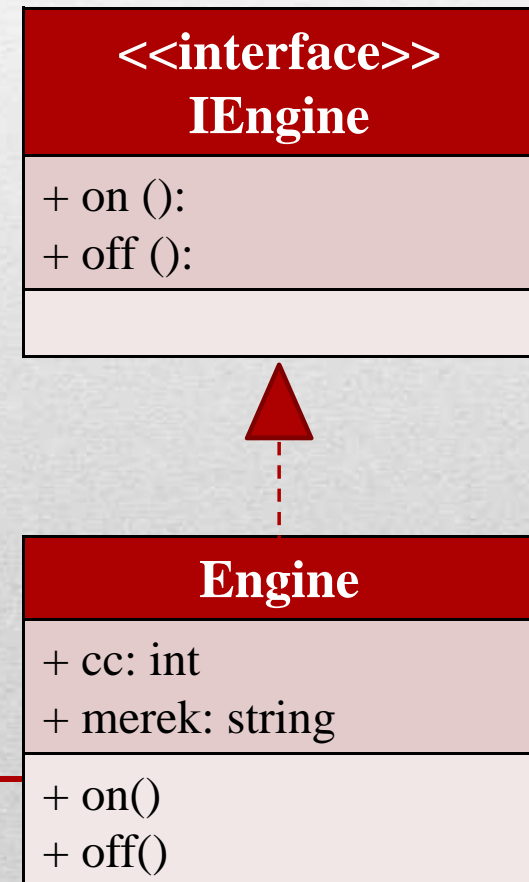
- Inheritance merupakan relasi turunan dimana sebuah kelas diciptakan berdasarkan kelas lainnya.

# Inheritance

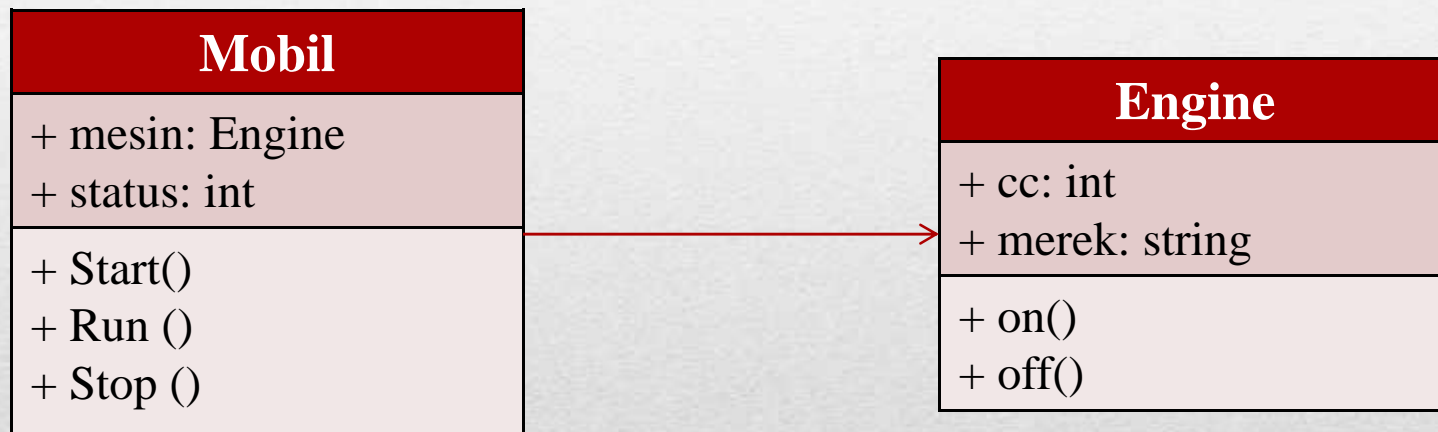


- Realization merupakan relasi yang terjadi akibat implementasi dari interface.
- Dalam relasi realization, sebuah kelas yang mengimplementasikan interface tertentu, harus mendefinisikan/mengimplementasikan seluruh method yang dideklarasikan dalam interface.

# Realization



- Dependency merupakan relasi antar kelas dimana satu kelas membutuhkan atau tergantung kepada kelas lainnya.

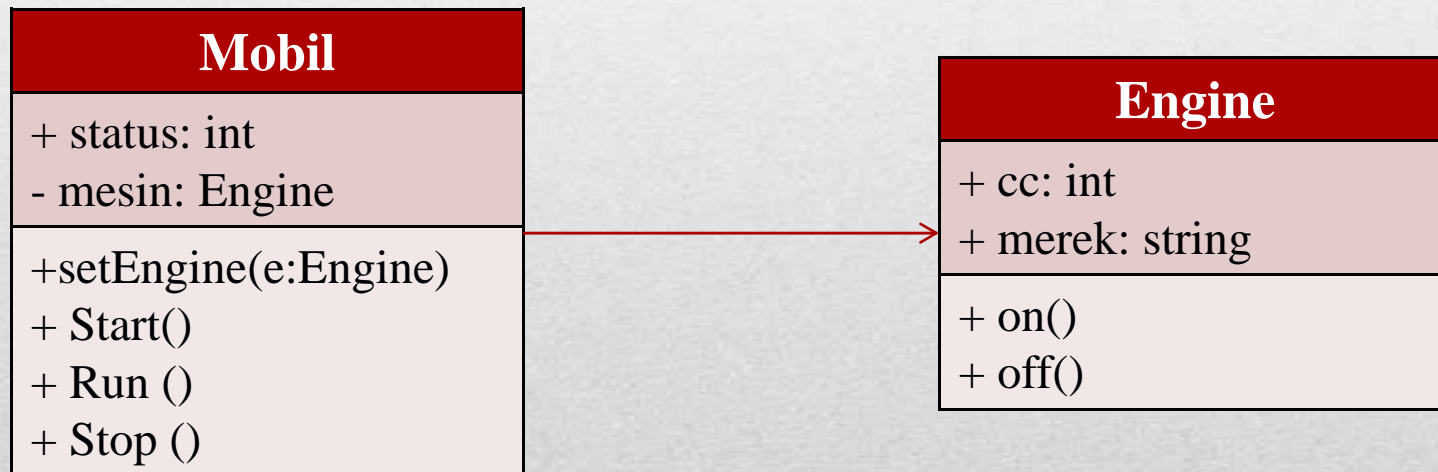


# Dependency

---



- Hubungan antar 2 kelas dimana kelas yg satu merupakan bagian dari kelas yg lain (kedua kelas dapat berdiri sendiri)
- Relasi aggregation merupakan bentuk khusus dari relasi dependency



# Agregation

---

```

class Engine
{
    public int cc;
    public String merek;

    public void On()
    {
        Console.WriteLine("Mesin ON");
    }

    public void Off()
    {
        Console.WriteLine("Mesin OFF");
    }
}

class Mobil
{
    private Engine mesin;
    public int status;

```

```

    public void setEngine(Engine e){
        mesin = e;
    }

    public void Start() {
        mesin.On();
    }
    public void Run() {
        Console.WriteLine("Run...!");
    }
    public void Stop() {
        mesin.Off();
    }
}

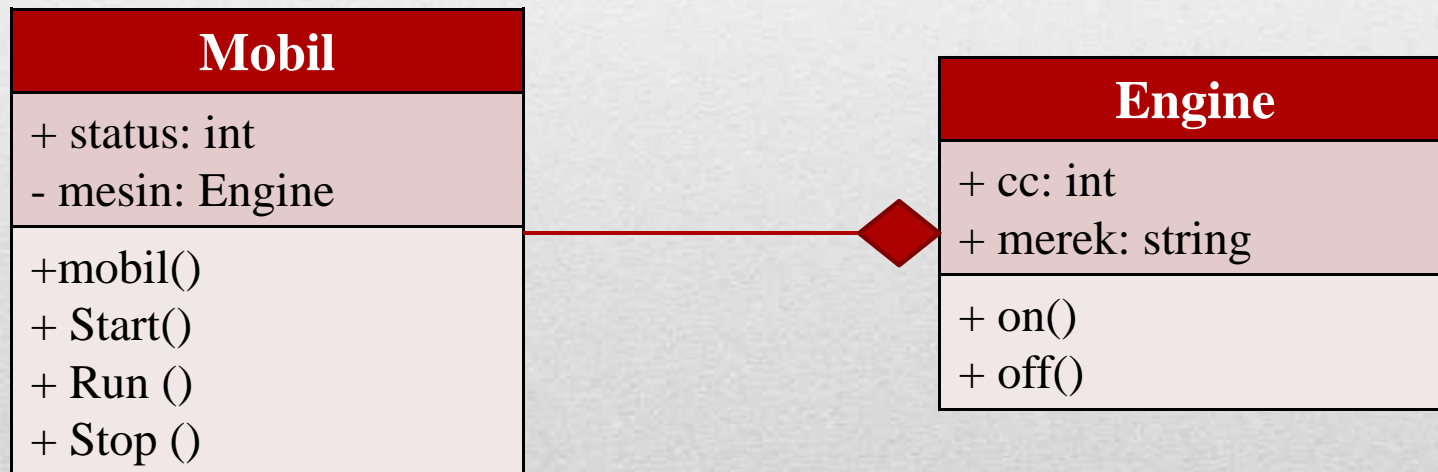
class Program
{
    static void Main(string[] args)
    {
        Engine engine = new Engine();
        Mobil mobil = new Mobil();
        mobil.setEngine(engine);
    }
}

```

# Aggregation

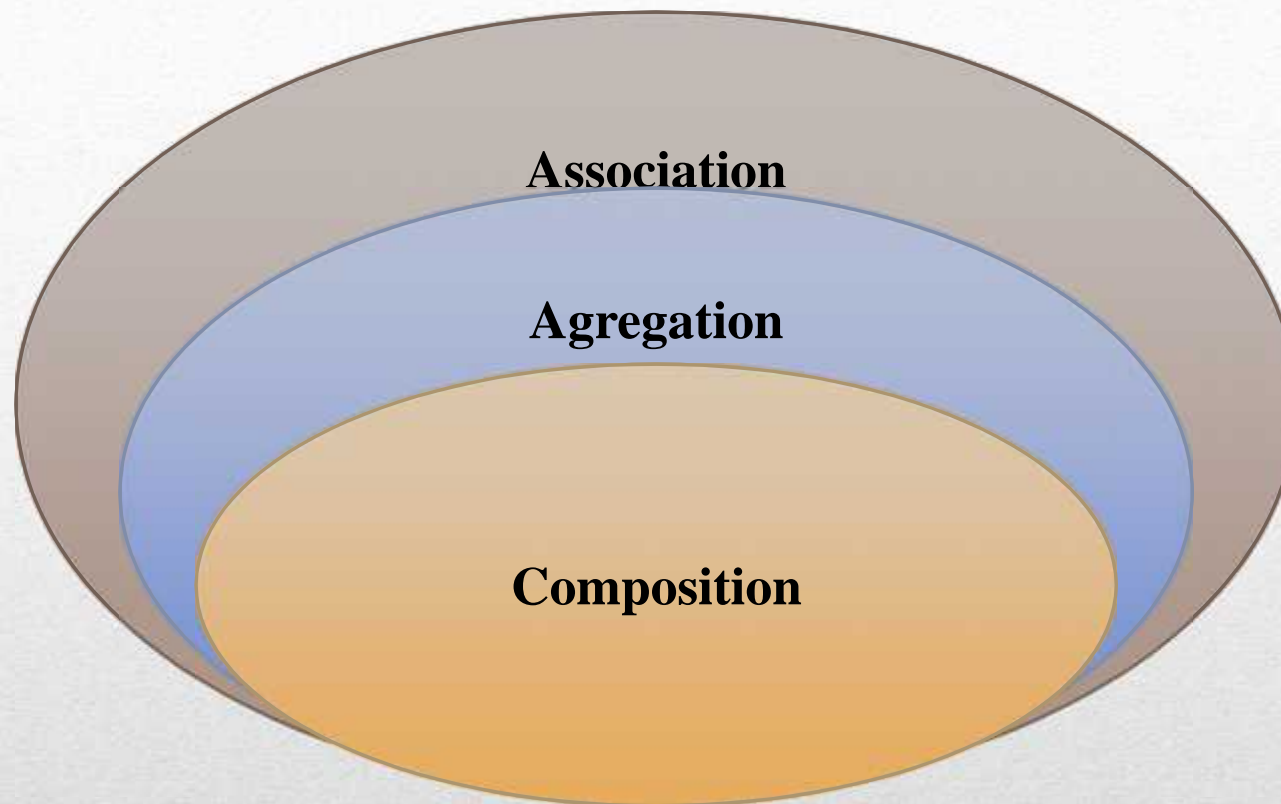
---

- Merupakan relasi yang lebih spesifik dari relasi aggregation
- Pada relasi ini suatu kelas tidak hanya dimiliki oleh kelas lainnya, tapi juga siklus hidup kelas tersebut juga ditentukan oleh kelas yang memilikinya.



# Composition

---



- Relasi composition merupakan bentuk khusus dari relasi aggregation dan relasi aggregation merupakan bentuk khusus dari relasi association.
-



