



PEMROGRAMAN JAVA

Yoannita



● **Konsep Pemrograman Berorientasi Objek**

“We have a hunger of the mind which asks for knowledge of all around us, and the more we gain, the more is our desire; the more we see, the more we are capable of seeing.”

(Maria Mitchell)

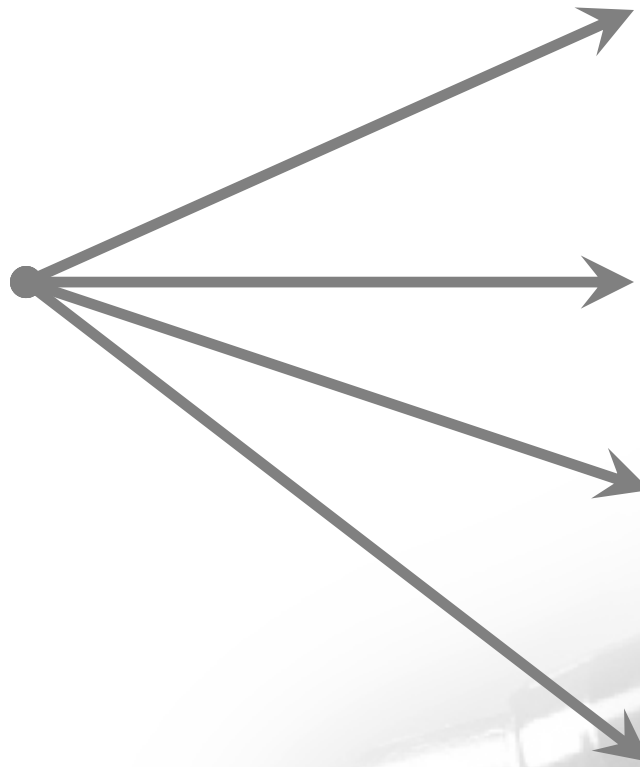
Konsep Pemrograman Berorientasi Objek

- **Class** adalah *blueprint* atau *prototype* dari objek-objek tertentu yang memiliki kesamaan variable dan method.
- Class merupakan *template* untuk sekumpulan objek dengan fitur yang sama.
- **Object** atau instance of class merupakan representasi nyata dari class.

Konsep Pemograman Berorientasi Objek



Class



Object

Konsep Pemograman Berorientasi Objek

- Class pohon → mendeskripsikan fitur yang dimiliki oleh semua pohon (memiliki akar, daun, tumbuh, dst)
- Class pohon berfungsi sebagai model abstrak tentang konsep pohon.
- Dari Class pohon tsb, anda dapat membuat berbagai pohon yang masing-masing bisa memiliki fitur berbeda (pendek, tinggi, berdaun lebat, dst) tetapi tetap dikenali sebagai pohon.
- Instance dari suatu class adalah kata lain dari objek aktual.
- Instance adalah representasi kongkrit dan spesifik dari kelas.
- Objek dan instance adalah sama.

Konsep Pemograman Berorientasi Objek

- Class Tombol
- Fitur tombol → label, ukuran, tampilannya,
- Perilaku → klik, doubleklik, warna berubah,
- Dengan membuat kelas Tombol, anda tidak perlu menulis ulang kode untuk tiap-tiap tombol yang anda pakai dalam program. Anda juga dapat menggunakan kembali kelas tombol untuk membuat jenis tombol yang lain untuk program yang sama maupun program lain.

Konsep Pemrograman Berorientasi Objek

Object Oriented Programming

- Istilah-istilah pada OOP :
 - State and behaviour
 - Encapsulation
 - Inheritance (Pewarisan)
 - Polymorphysm
- Saat sebuah objek dianalisa dan dikelompokkan, maka muncullah dua komponen utama dari sebuah objek, yaitu *state* dan *behaviour*. Serta tiga sifat utama yaitu *enkapsulasi*, *pewarisan*, dan *polymorphism*
- *Pemrograman berorientasi objek menggunakan model pembentukan sistem dimana komponen sistem (objek) seringkali terbentuk dari objek-objek lain yang lebih kecil.*

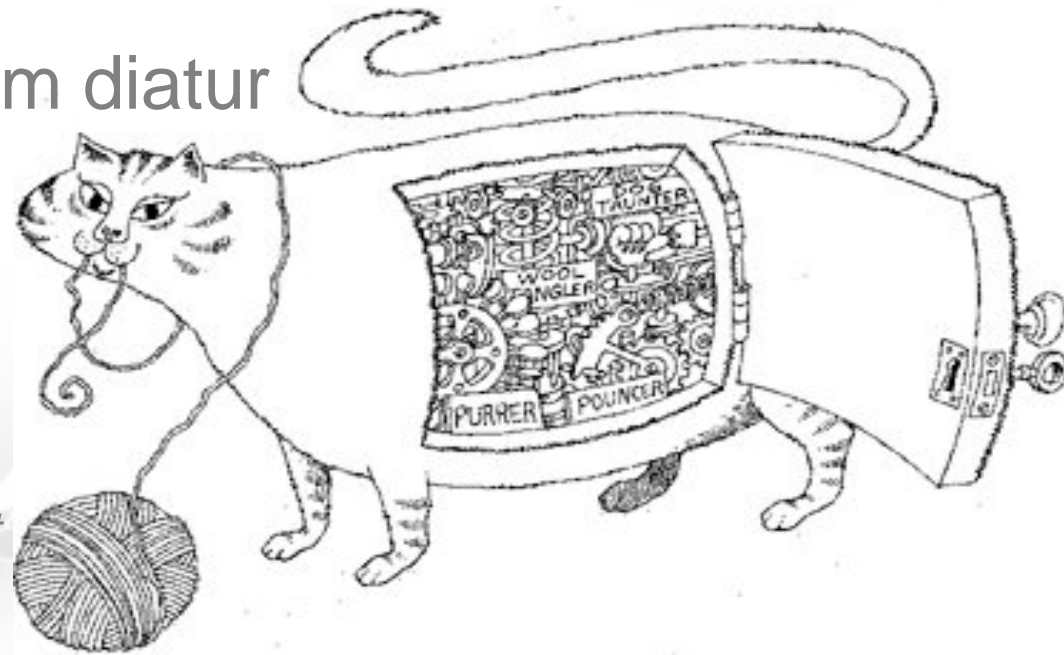
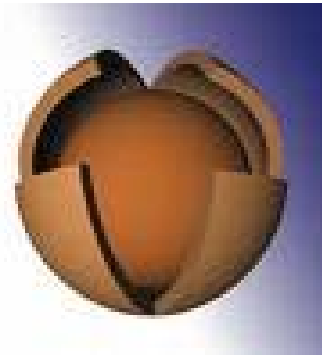


State and behaviour

- Setiap objek memiliki suatu keadaan (state) dan behaviour yang dapat mengubah state tersebut
- State merupakan suatu identitas dari objek
 - Setiap barang memiliki nama, harga, jenis, dst
 - Diimplementasikan sbg variabel atau field
- Behaviour dapat diartikan sebagai kegiatan dari objek.
 - Diimplementasikan dalam program sebagai proses/method
- State = kata benda, behaviour = kata kerja
- Contoh :
 - Manusia
 - State : umur, tinggi, berat badan
 - Behaviour : makan, tidur, bekerja

Enkapsulasi

- Suatu mekanisme untuk menyembunyikan atau memproteksi suatu proses dari kemungkinan interferensi atau penyalahgunaan dari luar sistem sekaligus menyederhanakan penggunaan sistem itu sendiri.
- Akses internal ke sistem diatur melalui *interface*



encapsulation hides the details of the implementation of an object

Enkapsulasi



Contoh :

- Sistem transmisi di dalam mobil menyembunyikan dari anda bagaimana cara ia bekerja, mulai dari bagaimana cara ia mengatur percepatan dan apa yang dilakukannya terhadap mesin mobil untuk mendapatkan percepatan tersebut.
- Anda sebagai pengguna hanya cukup memindah-mindahkan tongkat transmisi untuk mendapatkan percepatan yang diinginkan. Tongkat transmisi inilah yang menjadi satu-satunya interface dalam mengatur sistem transmisi dalam mobil tersebut.
- Kita tidak dapat menggunakan pedal rem untuk mengakses sistem transmisi tersebut. Sebaliknya, dengan mengubah transmisi tsb tidak akan dapat menghidupkan radio mobil atau membuka pintu mobil untuk anda

Enkapsulasi

Contoh lain :

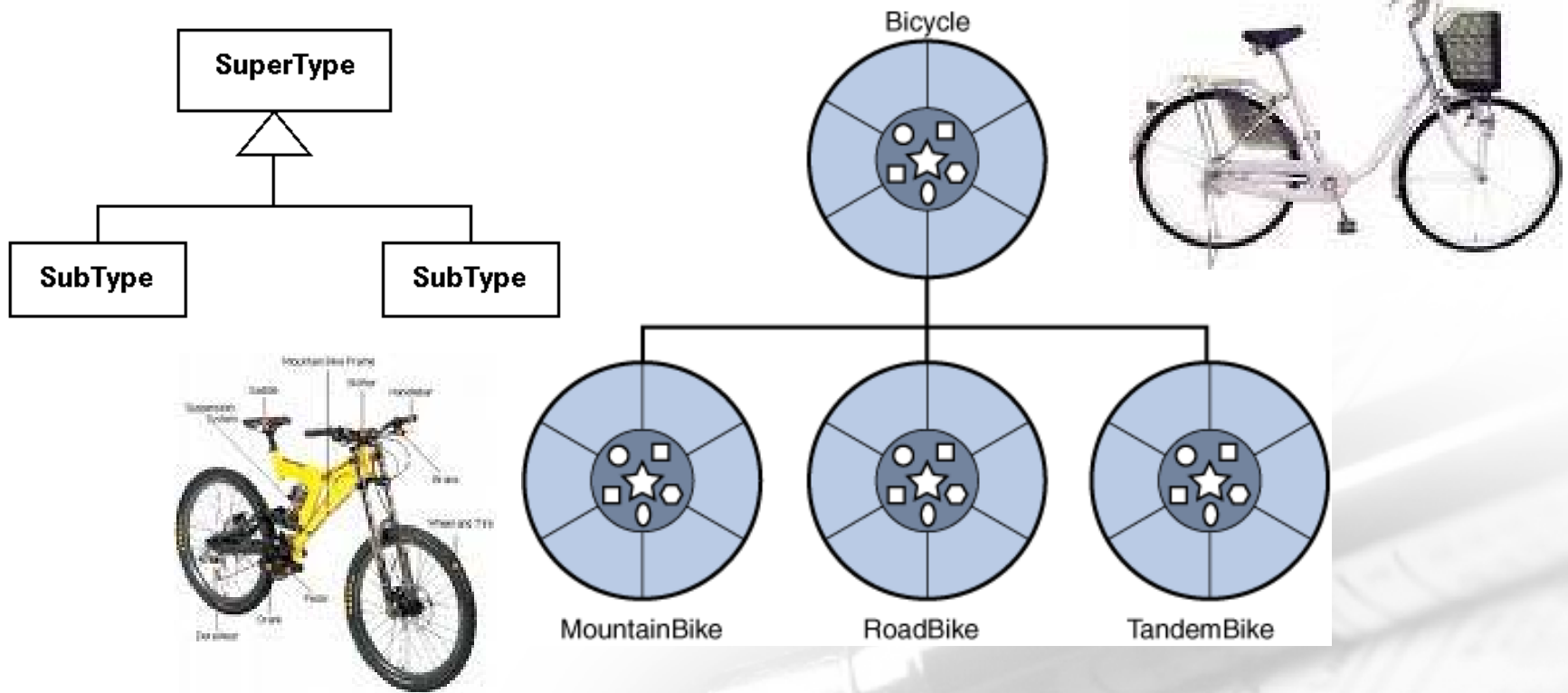
- Misalnya saja sistem pengeras suara pada radio dienkapsulasi tersendiri, sehingga jika Anda memindahkan gelombang radio, maka besar kecilnya suara tidak akan terpengaruh. Pemutar gelombang merupakan interface bagi Anda untuk mengubah gelombang radio.



Untuk menerapkan enkapsulasi pada Java, Anda cukup mendeklarasikan sebuah class, karena class merupakan dasar dari enkapsulasi. Setelah mendeklarasikan class, Anda tinggal mengisinya dengan state (variabel-variabel) dan behaviour (procedure / function). Serta sekaligus Anda dapat membuat state yang bersifat global (public) atau bersifat khusus (private).

Inheritance (Pewarisan)

- Apa yang terdapat pada super-class akan dimiliki juga oleh sub-class.



```
class MountainBike extends Bicycle {  
    // new fields and methods defining a mountain bike would go here  
}
```

Polymorphysm

- Konsep yang menyatakan bahwa **sesuatu yang sama** dapat mempunyai bentuk dan perilaku yang **berbeda**.
- Contoh : operasi move pada class graphic berbeda dengan move pada class mobil.



Abstraction
focuses on the
essential
characteristics
of some object,
relative to the
perspective

