Pemrograman Berorientasi Objek

Interface

Interface

Interface

- Interface adalah prototype kelas yang berisi definisi konstanta dan deklarasi method (hanya nama method tanpa definisi kode programnya).
- Interface digunakan untuk menyatakan spefisikasi fungsional beberapa kelas secara umum.

Interface

- Interface memungkinkan dibentuknya konsep pewarisan jamak (Multiple inheritance).
- Simbol relasi: realization—panah seperti inheritance, tapi putus-putus
- Simbol: sama seperti class, dengan *keyword* “interface” → <<>>

Interface

- Interfaces di-design untuk men-support *dynamic method resolution* saat run time.
- Interfaces men-disconnect definisi sebuah atau beberapa method dari hirarki pewarisan
- Dikarenakan interfaces tidak termasuk dalam hirarki inheritance, sebuah atau beberapa class yang tidak terkait memungkinkan untuk meng-implements interface tersebut.
  - Contoh: Listener pada Action Component (materi pemr. visual)
  - Contoh: Penggunaan thread

Content Interface

- Konstanta
- Method Abstract
- Tidak ada Konstruktor

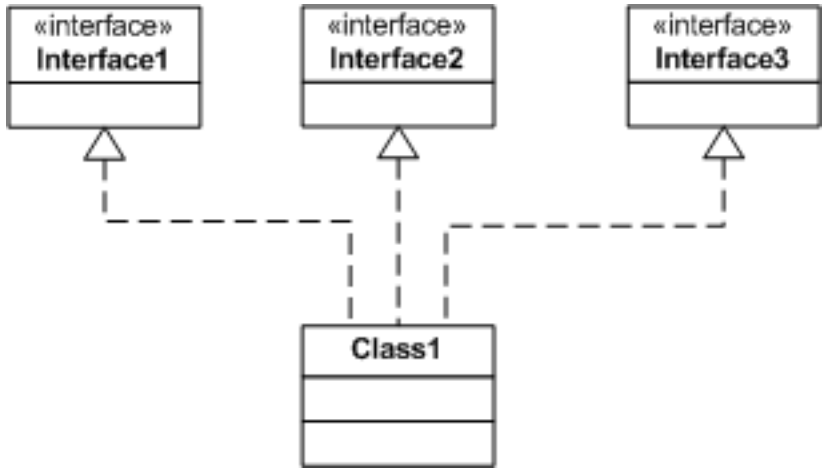
Aturan Interface

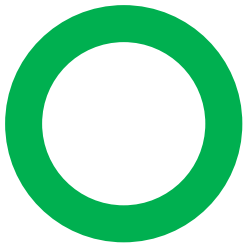
- Semua atribut adalah public, static dan final (semua atribut bertindak sebagai konstanta)
- Semua method adalah abstract dan public
- Tidak boleh ada deklarasi konstruktor

Aturan Pemakaian Interface

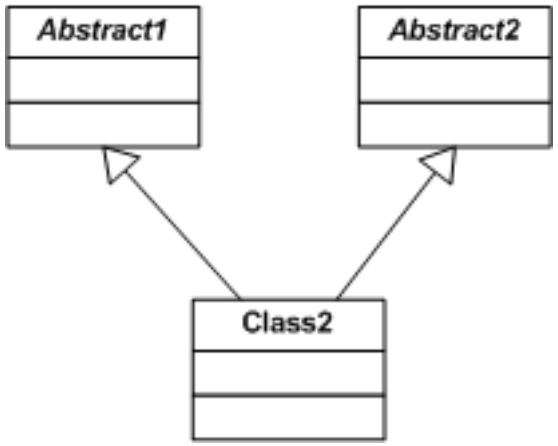
- Interface digunakan pada sebuah class dengan menggunakan keyword “implements”
- Sebuah class yang meng-implements sebuah interface, harus meng-override setiap method yang tercantum pada interface tersebut
- “*one interface, multiple methods*” aspek dari polymorphism.

IMPLEMENTS VS EXTENDS

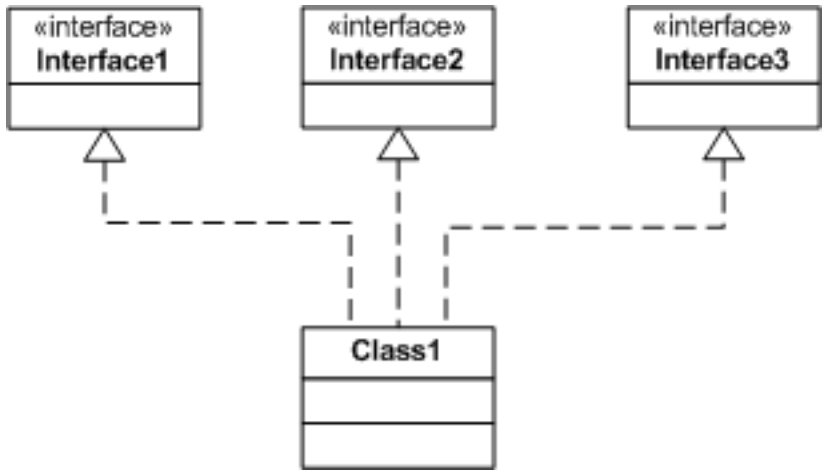




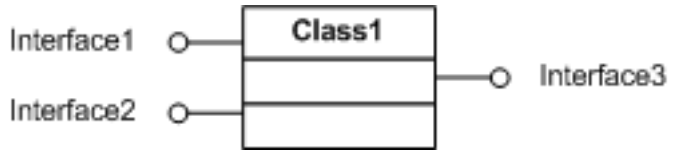
Interface memungkinkan
Pewarisan Jamak



Penggambaran Interface



Versi Lengkap



Versi Minimal

Contoh Interface

```
interface Callback {
  void callback(int param);
}
```

```
class TestIface {
  public static void main(String args[]) {
    Callback c = new Client();
    c.callback(42);
  }
}
```

```
class Client implements Callback {
  // Implement Callback's interface
  public void callback(int p) {
    System.out.println("callback called with " + p);
  }

  void nonIfaceMeth() {
    System.out.println("Classes that implement interfaces " +
                       "may also define other members, too.");
  }
}
```

Hasilnya??

Contoh Interface

```
interface Callback {
  void callback(int param);
}
```

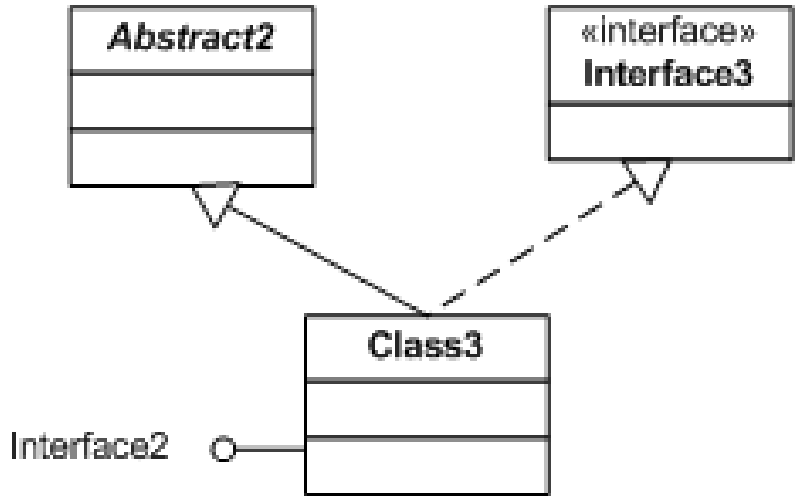
```
// Another implementation of Callback.
class AnotherClient implements Callback {
  // Implement Callback's interface
  public void callback(int p) {
    System.out.println("Another version of callback");
    System.out.println("p squared is " + (p*p));
  }
}
```

```
class TestIface2 {
  public static void main(String args[]) {
    Callback c = new Client();
    AnotherClient ob = new AnotherClient();

     c.callback(42);

     c = ob; // c now refers to AnotherClient object
     c.callback(42);
  }
}
```

The output from this program is shown here:

```
callback called with 42
Another version of callback
p squared is 1764
```

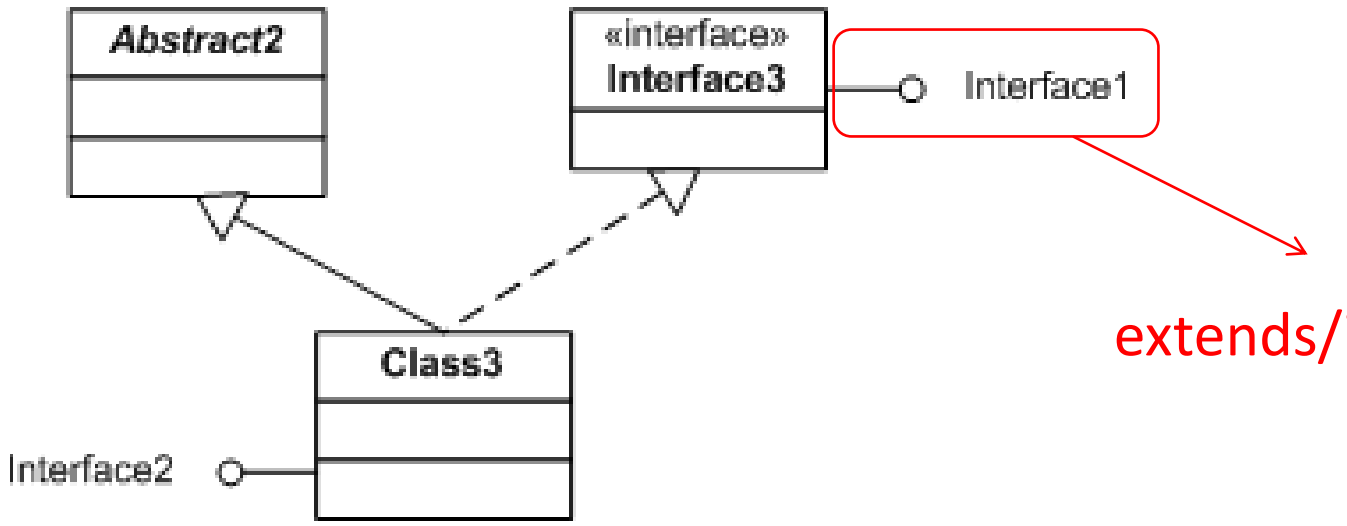
Kasus lain...



Bagaimana kodenya??

Bagaimana dengan method yang harus di-override?

Kasus lain...



extends/implements?

Bagaimana kodenya??

Bagaimana dengan method yang harus di-override?

Fungsi Abstract & Interface: Menerapkan Prinsip Design Pattern

Program to interface, not an implementation

Akhir Presentasi