Chapter FOUR

Interfaces

Like a class, an interface has either `public` or default accessibility:

```
public interface PublicAccessInterface {
    // ...
}
```

```
interface DefaultAccessInterface {
    // ...
}
```

Interfaces are `abstract` by default (you don't have to specify it):

```
public abstract interface PublicAccessInterface {
    // This is the same as the definition above
}
```

- You can't instantiate an interface directly, it must be implemented by a class to use it.
- An interface cannot be marked as final.

Method ใน interface เป็น public abstract โดยไม่ต้องมีการระบุ

Fields ใน interface เป็น public static final (CONSTANTS)

Interface สามารถ extends interface อื่นๆได้ และได้มากกว่า 1

Default methods

- Interface สามารถมี abstract และ default method เท่าไหร่ก็ได้
- default method ต้องมี body
- default method เป็น public โดยอัตโนมัติ
- The implemented classes can use and optionally redefine these methods.

```
interface Processable {
    void processInSequence();
    default void processInParallel() {
        /** Default implementation goes here */
    }
}
```

ข้อกำหนดของ default methods

1. Default methods cannot be final. เพราะจะ override ไม่ได้
2. Default methods cannot be synchronized.
3. Default methods are always public. but not static.
4. You cannot have default methods for the Object's class methods. A default method cannot override a method from java.lang.Object
    a. **boolean** equals(Object o);// not require implements

b. `int hashCode();// not require implements`

c. `String toString();// not require implements`

More specific interfaces (or classes) always WIN over less specific ones

```
interface Processable {
    void processInSequence();
    default void processInParallel() {
        System.out.println("Processable parallel");
    }
}
interface Parallelizable extends Processable {
    default void processInParallel() {
        System.out.println("Parallelizable parallel");
    }
}
public class Task implements Parallelizable {
    public void processInSequence() {
        System.out.println("Processing in sequence");
    }
    public static void main(String args[]) {
        Task t = new Task();
        t.processInParallel();
    }
}
```

The output would be:

```
Parallelizable parallel
```

วิธีการเรียกใช้ Default method ใน Interface

```
NameOfTheInteface.super.defaultMethod();
```

Multiple interface inheritance with default methods

- Classes can implement multiple interfaces. ถ้ามี default method ชื่อเหมือนกันจะ compile error แต่สามารถแก้ได้โดยทำการ override method นั้นมาใช้

```
interface Processable {
    void processInSequence();
    default void processInParallel() {
        System.out.println("Processable parallel");
    }
}
interface Parallelizable {
    default void processInParallel() {
        System.out.println("Parallelizable parallel");
    }
}
public class Task
            implements Processable, Parallelizable {
    public void processInSequence() {
        System.out.println("Processing in sequence");
    }
    public static void main(String args[]) {
        Task t = new Task();
        t.processInParallel();
    }
}
```

It turns out that the result is a compiler error:

```
Duplicate default methods named processInParallel with the parameters () and ()
```

Static method เป็น method ที่ใช้ร่วมกันระดับ class

โดย Static method ต้องมี body และไม่สามารถสืบทอดคุณสมบัติได้

วิธีการเรียกใช้ Static method

```
NameOfTheInteface.staticMethod();
```

ตัวอย่าง

```
interface Parallelizable {
    static void log(String s) {
        System.out.println(s);
    }
    default void processInParallel() {
        log("Parallelizable parallel");
    }
}
public class Task implements Parallelizable {
    public static void main(String args[]) {
        Task t = new Task();
        t.processInParallel();
        // t.log("The end"); Doesn't compile
        // Task.log("The end"); Doesn't compile either
        Parallelizable.log("The end"); // Compiles!
    }
}
```

The output is:

```
Parallelizable parallel The end
```