More on Abstraction in Java

Introduction to Interfaces

Produced Dr. Siobhán Drohan by: Mairead Meagher Eamonn de Leastar



Waterford Institute *of* Technology

Department of Computing and Mathematics http://www.wit.ie/

Topic List

– What are Interfaces?

- Syntax for an Interface.
- Implementing Interfaces.

Interfaces

- We now know why multiple inheritance is not allowed in Java.
- However, there is a way to "simulate" multiple inheritance.
- We will now look at interfaces which are used when you can see a "multiple inheritance" in your class design.

What is an interface?

• Writing an interface is similar to writing a class.

 But a class describes the attributes and behaviours of an object.

And an interface contains behaviours that a class implements.

What is an interface?

- An interface is:
 - a reference type in Java
 - similar(ish) to a class,
 - a collection of abstract method signatures.
- A class implements an interface, thereby inheriting the abstract methods of the interface.

What is an interface?

- Along with abstract methods an interface may also contain:
 - constants i.e. final static fields
 - default methods
 - static methods
- Method bodies exist <u>only</u> for default methods and static methods.
- NOTE: Pre Java 8, Interfaces did not have static and default methods.

Interface Rules Summary

- Interfaces can contain:
 - Only method signatures for abstract methods.
 - Only final static fields.
 - default and static methods (including their implementation).
- Interfaces cannot contain:
 - Any fields other than public final static fields.
 - Any constructors.
 - Any concrete methods, other than default and static ones.

Topic List

- What are Interfaces?

- Syntax for an Interface.

- Implementing Interfaces.

• Writing an interface is similar to writing a class.

- But...
 - a class describes the attributes and behaviours of an object.
 - an interface contains behaviours that a class implements.

• The interface keyword is used to declare an interface.

 Unless the class that implements the interface is abstract, <u>all</u> the abstract methods of the interface need to be defined in the class.

import java.lang.*;
//Any number of import statements

File name : NameOfInterface.java

public interface NameOfInterface {

}

//Any number of final, static fields
//Any number of abstract method declarations
//Any number of default and static method implementations

import java.lang.*;
//Any number of import statements

File name : NameOfInterface.java

public interface NameOfInterface {

//Any number of final, static fields
//Any number of abstract method declarations
//Any number of default and static method implementations

```
interface IMammal
{
    public void eat();
```

}

File name : IMammal.java

public void eat(); public void travel();

Interface IMammal

{

}

void eat();
void travel();

- Interfaces have the following properties:
 - An interface is implicitly abstract. You do not need to use the **abstract** keyword while declaring an interface.
 - Each method in an interface is also implicitly abstract, so the abstract keyword is not needed.
 - Methods in an interface are implicitly public, so the keyword public is also not required.

Topic List

- What are Interfaces?

- Syntax for an Interface.

- Implementing Interfaces.

- When a class implements an interface:
 - you can think of the class as signing a contract, agreeing to perform the specific behaviours of the interface.
- If a class does not perform all the behaviours of the interface, the class must declare itself as abstract.

• A class uses the **implements** keyword to implement an interface.

• The implements keyword appears in the class declaration following the extends portion (if there is one).

```
public class Mammal implements IMammal{
    public void eat(){
        System.out.println("Mammal eats");
    }
```

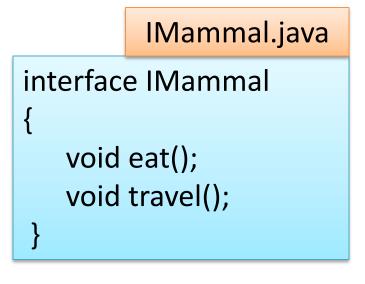
```
public void travel(){
    System.out.println("Mammal travels");
}
```

```
public int noOfLegs(){
    return 0;
```

```
}
```

```
public static void main(String args[]){
    Mammal m = new Mammal();
    m.eat();
    m.travel();
```

```
Mammal.java
```





- When implementation interfaces there are several rules:
 - A class can implement more than one interface at a time.
 - A class can extend only one class, but implement many interfaces.
 - An interface can extend another interface, similarly to the way that a class can extend another class.
 - An interface cannot implement another interface.

Any Questions?





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see http:// creativecommons.org/licenses/by-nc/3.0/



Waterford Institute *of* Technology

Department of Computing and Mathematics http://www.wit.ie/