Lesson 13-2: Interfaces

Computer Science 46A: Introduction to Programming

San José State University

Announcements

- Homework8 Posted. This is labeled homework 12 in Google Drive

- Lab this Friday (5/2)

- Midterm 2 on May 6
  - Some example test bank questions have been posted
  - Same format as first mid term

Learning Objectives

By the end of this lesson, you should be able to:

1. Describe how interfaces are used in Java

2. Implement the Comparable interface and override the default compareTo method

3. Implement a new interface to structure a framework for classes with similar behavior

\# Abstract methods. (only prototype of method, means
it has no body).

\# the variable are by default public static.
and final. (constants).

The Comparable Interface

public interface Comparable {
    public int compareTo (Object ob);

        This is
        abstract
        method.

    int max = Integer. MAX_VALUE;

}

A Motivating Example: comparing objects

In Java, we can compare objects to determine whether an object is ordered before, after or at the same place as another object:

- Integers and Doubles: comparison determines whether a number comes before or after on the number line (>, =, or <)
- Characters: comparison determines where the letter is in the alphabet
- Strings: comparison determines the ordering of the string in alphabetical order (like Webster's dictionary)

See example codes in **Comparisons** in lec13-2a

The `compareTo()` method

- Most default classes implement a compareTo method that is used for comparisons:

$$a.compareTo(b) > 0$$

- Method returns:
  - <0 if a comes before b
  - >0 if a comes after b
  - 0 if a and b are in same place
- All default classes implement the compareTo method, but the way they operate depends on the object

See example codes in **Comparisons** in lec13-2a
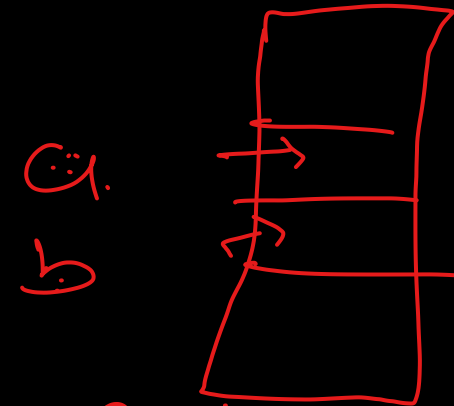
Interfaces Defined

- An <u>interface</u> defines methods that are *required* within a given class that implements the method
  - All classes that implement the interface can be used in the same algorithm
- Comparison Example: when classes implement the Comparison interface, they are required to have a compareTo() method

public class XYZ implementsComparable}
@ Override.
Public int compareTo (Object ob)}
step-1 convert 'ob' in your requiredType.

and then proceed.

Implementing an Interface

- To implement an interface for a class, we use a syntax similar to the extension of a superclass to a subclass

    ```
    public class [classname] implements [interface name]
    ```

- For example, to implement the Comparable interface in the Car class, we use the statement:

    ```
    public class Car implements Comparable<Car>
    ```

See example codes in **CarTester** in lec13-2b

Using the compareTo method: Sorting

- Comparisons can be used to sort an ArrayList
- The Collections package has a method sort() that leverages the compareTo method to arrange objects in ascending order
- Objects need a compareTo method in order to use the sort method
- compareTo methods use @Override

See example codes in **IntListTester** and **CarListTester** in lec13-2c

Poll Everywhere: Question 1

Which of the following terms describe the sort() method from the Collections class?

A) Accessor

B) Mutator

C) Static

D) B and C

E) A and C

Collections.sort()

↑ class reference.   ↑ static

General Interfaces

When To Use An Interface

- Interfaces are created to provide a method framework across classes
- When similar classes need to have the same method to be used in the same algorithm
  - ⋯→ implement an interface

circle

Rectangle

Line

draw ( )

draw ( )

draw ( )

Declaring an Interface

- Interfaces are declared just like a class:
  ```
  public interface [interface name]
  {
      method1
      method2

      …
  }
  ```

- Interfaces can best be thought of as frameworks

- Interfaces do not have instance variables

Interface Methods

Methods in an interface as declared as:

`[output type]  [method name]();`

Interfaces declare the methods that are required in the class implementations

- Interface methods do not contain a body!

- All interface methods are public (no need to add the keyword)

Example: ArrayListUtils

- Goal: Create a class with a method that will return the biggest object in an ArrayList
  - Method should work for *any* objects that are put into in the ArrayList
- In this example, we implement an interface called Measurable
  - The Measurable interface defines a framework to generate a getMeasure method across different classes
  - Any class that uses the Measurable interface can be used in the same way
  - See the biggest[class] functionality in ArrayListUtilsTester

Compare codes for **ArrayListUtilsTester** in lec13-2d

Interfaces vs Inheritance

Interfaces are not the same as Inheritance

Inhertiance

- Objects from subclasses of the same superclass share common instance variables and methods

- Inherited methods are exactly the same across subclasses

Interfaces

- Objects from classes that implement the same interface do not share any variables or methods

- Interface methods function similarly, but are specific to the class that implements them

Participation Exercise 13-2a: Qualifiable

<u>Goal:</u> Implement the `qualifies()` method from the Qualifiable interface in the Student and BankAccount classes

```
Testing BankAccount
true
Expected: true
false
Expected: false
false
Expected: false
true
Expected: true
Testing Students
false
Expected: false
true
Expected: true
true
Expected: true
```

Expected output from the InterfaceTester class

Codecheck Link: HERE and on Canvas

Participation Exercise 13-2b: Stock

Goal: Implement the Comparable interface in the Stock class (from Homework 8) and generate a new compareTo methods for stocks.

Then, write a sort() method in the StockList class that will sort stocks based on your new compareTo method

```
Before Sort:
[Stock[symbol=GOOGL,price=1757.76], Stock[symbol=AMZN,price=1757.76], Stock[symbol=FB,price=276.78]]
Expected: [Stock[symbol=GOOGL,price=1757.76], Stock[symbol=AMZN,price=1757.76], Stock[symbol=FB,price=276.78]]

After Sort:
[Stock[symbol=FB,price=276.78], Stock[symbol=AMZN,price=1757.76], Stock[symbol=GOOGL,price=1757.76]]
Expected: [Stock[symbol=FB,price=276.78], Stock[symbol=AMZN,price=1757.76], Stock[symbol=GOOGL,price=1757.76]]
```

Expected output from the StockListTester class

Codecheck Link: HERE and on Canvas