

CS151 Lab#4: Interface

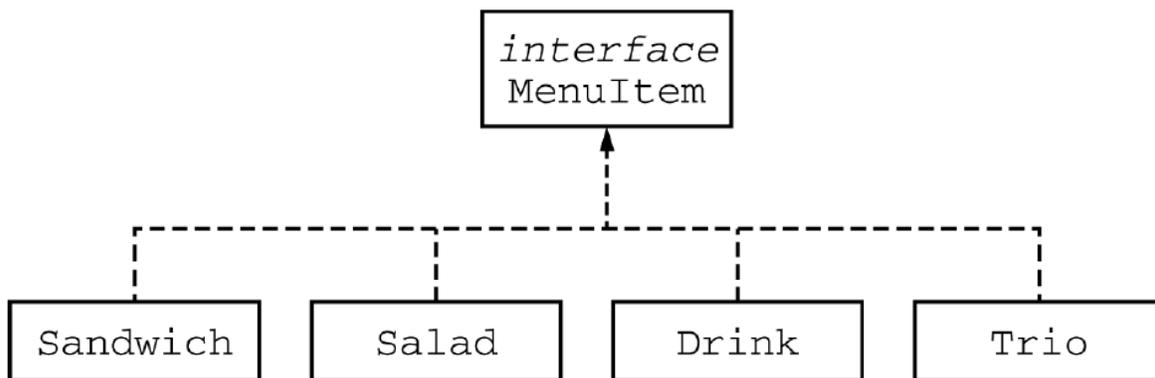
Lab exercises due by the assignment 4 due date.

In this lab, we will learn about Java interfaces. Look in `~dxu/handouts/cs151/labs/04` for code related to this lab.

The menu at a lunch counter includes a variety of sandwiches, salads, and drinks. The menu also allows a customer to create a “trio,” which consists of three menu items, one of each category: a sandwich, a salad, and a drink. Each menu item has a name and a price. The four types of menu items are represented by the four classes `Sandwich`, `Salad`, `Drink`, and `Trio`. All four classes implement the following `MenuItem` interface.

```
public interface MenuItem {  
    /**  
     * @returns the name of the menu item  
     */  
    String getName();  
  
    /**  
     * @return the price of the menu item  
     */  
    double getPrice();  
}
```

The following diagram shows the relationship between the `MenuItem` interface and the `Sandwich`, `Salad`, `Drink`, and `Trio` classes.



For example, assume that the menu includes the following items. The objects listed under each heading are instances of the class indicated by the heading.

Sandwich	Salad	Drink
"Cheeseburger" 2.75	"Spinach Salad" 1.25	"Orange Soda" 1.25
"Club Sandwich" 2.75	"Coleslaw" 1.25	"Cappuccino" 3.50

The name of the Trio consists of the names of the sandwich, salad, and drink, in that order, each separated by "/" and followed by a space and then "Trio". The price of the Trio is the sum of the two highest-priced items in the Trio; one item with the lowest price is free. A trio consisting of a cheeseburger, spinach salad, and an orange soda would have the name "Cheeseburger/Spinach Salad/Orange Soda Trio" and a price of \$4.00 (the two highest prices are \$2.75 and \$1.25). Similarly, a trio consisting of a club sandwich, coleslaw, and a cappuccino would have the name "Club Sandwich/Coleslaw/Cappuccino Trio" and a price of \$6.25 (the two highest prices are \$2.75 and \$3.50).

Exercise 1: Implement the Sandwich, Salad and Drink classes as specified. Test with the following driver program:

```
public static void main(String[] args) {
    Sandwich burger = new Sandwich("Cheeseburger",2.75);
    Sandwich club = new Sandwich("Club Sandwich", 2.75);
    Salad spinachSalad = new Salad("Spinach Salad",1.25);
    Salad coleslaw = new Salad("Coleslaw", 1.25);
    Drink orange = new Drink("Orange Soda", 1.25);
    Drink cap = new Drink("Cappuccino", 3.50);
    System.out.println(burger.getName()+ " " + burger.getPrice());
    System.out.println(club.getName()+ " " + club.getPrice());
    System.out.println(spinachSalad.getName()+ " " +
        spinachSalad.getPrice());
    System.out.println(coleslaw.getName()+ " " +
        coleslaw.getPrice());
    System.out.println(orange.getName()+ " " + orange.getPrice());
    System.out.println(cap.getName()+ " " + cap.getPrice());
}
```

Exercise 2: Implement the Trio class as specified. Test with the following driver program:

```
public static void main(String[] args) {
    Sandwich burger = new Sandwich("Cheeseburger",2.75);
    Sandwich club = new Sandwich("Club Sandwich", 2.75);
    Salad spinachSalad = new Salad("Spinach Salad",1.25);
    Salad coleslaw = new Salad("Coleslaw", 1.25);
    Drink orange = new Drink("Orange Soda", 1.25);
    Drink cap = new Drink("Cappuccino", 3.50);
```

```
Trio trio1 = new Trio(burger, spinachSalad, orange);
System.out.println(trio1.getName());
System.out.println(trio1.getPrice());
Trio trio2 = new Trio(club, coleslaw, cap);
System.out.println(trio2.getName());
System.out.println(trio2.getPrice());
}
```

Exercise 3: Modify the `Trio` class so that it throws a `IllegalTrioException` when anyone attempts to create a `Trio` combining three items of the same price (I don't know why, because giving away a third that's not "cheaper" isn't allowed? – no, I just needed to make up a reason for you to custom-design an exception!). Provide code in your driver to test it.

Exercise 4: Modify the `Trio` class to also implement `Comparable`. The ordering of the `Trios` depends on their prices – the more expensive `Trio` is "larger". Provide code in your driver to test it. Feel free to make up lots more sandwiches, salads and drinks to have some variety. Putting a bunch of `Trios` in an array and then call `Array.sort` on them is a good way to test.