INTERFACES: the Comparable interface

Samuel L. Marateck

An interface does not contain complete methods just their signatures (also known as prototypes or headings). It also can contain fields. An interface differs from a superclass in that a class can inherit from just one superclass but can implement more than one interface. You do this by using the word implements followed by the interface’s name, for instance, public class Huffman implements Comparable.

Another reason for using an interface’s is that using one enables you to make an operation, class independent. In other words, any class that implements the interface can be used in the operation. Let’s see how this is done using the Comparable interface. This interface contains the heading for only one method, compareTo. We have seen this method used with strings. For instance, if line1 and line2 are String references, then line2.compareTo(line1) compares the two strings in the dictionary sense. In the general case, the comparison is based on what is called the class’s natural ordering. Since the String class implements the Comparable interface, it must contain a compareTo method. All the wrapper classes except the Boolean one implement the Comparable interface. It is useful to create an array of a class that implements an interface. For example, Huffman x = new Huffman[20]; creates an array x of Huffman where Huffman implements the Comparable interface. Thus to determine if x[i] is greater than x[j], you would write x[i].compareTo(x[j]). You could use this to write a sorting routine, for example,

public void sort( Comparable[] x)
{ ...if( x[i].compareTo(x[i+1])...}

Now if you pass to the sort method an array of any class that implements Comparable, because of polymorphism, the compareTo method for that class is used at run-time. Remember, only those references of classes that implement Comparable, can be used here.