foreach loop

The foreach loop, or *enhanced for statement*, as Java calls it, is used to enumerate the values in a collection of values. We give an example.

The two loops shown below store in variable sum the sum of the elements of array b. The loop on the right is the usual for-loop. The foreach loop on the left is syntactic sugar for the loop on the right. It looks a lot simpler! The foreach loop will actually be converted into something like the loop on the right as the program is being compiled.

```
int sum= 0;
for (int v : b) {
   sum= sum + v;
}
```

```
k= 0;
int sum= 0;
while (k < b.length) {
  int v= b[k];
  sum= sum + v;
  k= k+1;
}
```

Here's another example. Suppose Set variable s contains a set of Integers. It might have been defined using

```
Set<Integer> s= new ArrayList<Integer>();
```

The following loop sums the values in s.

```
int sum= 0;
for (Integer v : s)  sum= sum + v;
```

These two examples illustrate the purpose of the foreach loop: to enumerate each element v of a collection and process it in some fashion. In each iteration of the foreach loop, the repetend is executed with v containing another value of the collection. For arrays and lists of any kind, the values are enumerated in the order in which they occur in the array or list. For other collections, like sets, you will probably not know the order.

We could have written the last foreach using type int instead of integer (shown below); then, each ArrayList element will be auto-unboxed before being stored in v:

```
for (int v : s)  sum= sum + v;
```

Syntax of a foreach loop

The syntax of a foreach loop should be clear from the examples. Between "(" and ":", place a declaration of a variable; its type is the type of the elements of the collection. Between ":" and ")", put an expression that gives the collection to be enumerated. As usual, the repetend is any statement.

On what collections can you use a foreach loop?

You can use the foreach statement on any array. In addition, each of the following classes and interfaces in package java.util supports the foreach statement (there may be others).

Set	List	Stack
SortedSet	ArrayList	Queue
HashSet	LinkedList	Dequeu
EnumSet	Vector	ArrayDeque
LinkedHashSet		

Fix your own collection to allow the foreach loop

Suppose you have written your own class to implement some collection of objects. To be able to use a foreach loop on that class, just implement interfaces java.util.Iterator and java.lang.Iterable!

This topic is discussed in detail in a tutorial on these two interfaces, with less than 15 minutes of video. See the entry for *foreach loop* in the list of concepts and definitions for a link to the tutorial.