Factories build things.

In OO parlance, a factory method returns an object of a class, building it (using a new-expression) or returning an already existing object. We explain and give examples.

**Class Integer**

Generally, there is no need for more than one object of class Integer that wraps 0, since the object is immutable. Yet, every evaluation of new Integer(0) creates a new object that wraps 0. Further, creating an object *does* take time. Therefore, since Java version 9, the constructors of class Integer have been deprecated, and this comment appears in the discussion of its constructor in the Java API:

... it is rarely appropriate to use this constructor. The static factory valueOf(int) is generally a better choice, as it is likely to yield significantly better space and time performance.

Here is part of the spec of method valueOf(int):

... this method is likely to yield significantly better space and time performance by caching frequently requested values. This method will always cache values in the range -128 to 127 ...

How to suppress the use of a constructor and write a factory method.

Make the constructor private! Then, only methods in the class can use the

new-expression. Then, write a public static method, with as many parameters as you

need, that returns an object of the class. You get to decide, based on the

parameters, whether a new object has to be created or whether an existing one

can be returned..

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.