Special Topic 15.1

The `Iterable` Interface and the "For Each" Loop

You can use the "for each" loop

```
for (Type variable : collection)
```

with any of the collection classes in the standard Java library. This includes the `ArrayList` and `LinkedList` classes as well as the library classes which will be discussed in Chapter 16. In fact, the "for each" loop can be used with any class that implements the `Iterable` interface:

```
public interface Iterable<E>
{
   Iterator<E> iterator();
}
```

The interface has a type parameter `E`, denoting the element type of the collection. The single method, `iterator`, yields an object that implements the `Iterator<E>` interface. That interface has methods

```
boolean hasNext();
E next();
```

The `ListIterator` interface that you saw in the preceding section is a subinterface of `Iterator` with additional methods (such as `add` and `previous`).

The compiler translates a "for each" loop into an equivalent loop that uses an iterator. The loop

```
for (Type variable : collection)
   body
```

is equivalent to

```
Iterator<Type> iter = collection.iterator();
while (iter.hasNext())
{
```

```
    Type variable = iter.next();
    body
}
```

The `ArrayList` and `LinkedList` classes implement the `Iterable` interface. If your own classes implement the `Iterable` interface, you can use them with the "for each" loop as well—see Exercise P15.19.