8.12 `static` Import

In Section 6.3, you learned about the `static` fields and methods of class `Math`. We access class `Math`'s `static` fields and *methods* by preceding each with the class name `Math` and a dot (`.`). A `static` **import** declaration enables you to import the `static` members of a class or interface so you can access them via their *unqualified names* in your class—that is, the class name and a dot (`.`) are *not* required when using an imported `static` member.

`static` Import Forms

A `static` import declaration has two forms—one that imports a particular `static` member (which is known as **single** `static` **import**) and one that imports *all* `static` members of a class (known as `static` **import on demand**). The following syntax imports a particular `static` member:

```
import static packageName.ClassName.staticMemberName;
```

where *packageName* is the package of the class (e.g., `java.lang`), *ClassName* is the name of the class (e.g., `Math`) and *staticMemberName* is the name of the `static`

field or method (e.g., `PI` or `abs`). In the following syntax, the asterisk (`*`) indicates that *all* `static` members of a class should be available for use in the file:

```
import static packageName.ClassName.*;
```

`static` import declarations import *only* `static` class members. Regular `import` statements should be used to specify the classes used in a program.

Demonstrating `static` Import

Figure 8.14 demonstrates a `static` import. Line 3 is a `static` import declaration, which imports *all* `static` fields and methods of class `Math` from package `java.lang`. Lines 7–10 access the `Math` class's `static` methods `sqrt` (line 7) and `ceil` (line 8) and its `static` fields E (line 9) and `PI` (line 10) *without* preceding the field names or method names with class name `Math` and a dot.

Common Programming Error 8.7

A compilation error occurs if a program attempts to import two or more classes' `static` *methods that have the same signature or* `static` *fields that have the same name.*

```
1   // Fig. 8.14: StaticImportTest.java
2   // Static import of Math class methods.
3   import static java.lang.Math.*;
4
5   public class StaticImportTest {
6      public static void main(String[] args) {
7         System.out.printf("sqrt(900.0) = %.1f%n",
8         System.out.printf("ceil(-9.8) = %.1f%n", c
9            System.out.printf("E = %f%n", E);
10          System.out.printf("PI = %f%n", PI);
11       }
12   }
```

```
sqrt(900.0) = 30.0
ceil(-9.8) = -9.0
   E = 2.718282
  PI = 3.141593
```

Fig. 8.14

`static` import of `Math` class methods.