Errata for Effective Java, Third Edition

Please visit this Google Drive URL for updated errata from the author.

Fixed in second printing:

p. xvii, par. 3: The comma after “Yoshiki Shibata” should be a period.

p. 8, par. 4: In the first sentence, the word “produce” should be “produces.”

p. 75, par. 3: The reference to "Item 15" should be replaced by "Item 10."

p. 164, par 3: In the first sentence, the type should be changed from `Optional<String>` to `Optional<Operation>`.

p. 213, code ex. 1, Changed lambda param names `(v1, v2)` to `(oldVal, newVal)` for clarity.

p. 322, par. 4: The reference “Herlihy08” should read “Herlihy12.”

p. 370, second ed. item numbers 69 and 74: Third ed. item numbers 81 and 86 are duplicated.

Fixed in third printing

p. viii, Item 47: Typewriter font should be used for the types `Collection` and `Stream`.

p. 18, par. 2: The method reference `elvis::instance` should read `elvis::getInstance`.

p. 184, code ex. 1:

```
    System.out.println("Invalid @Test: " + m);
```

should read:

```
    System.out.println("Invalid @ExceptionTest: " + m);
```

p. 184, code ex. 2: For consistency with the preceding example (and greater generality), the sole attribute declaration in the annotation declaration, which currently reads:

```
    Class<? extends Exception>[] value();
```

should read:

```
     Class<? extends Throwable>[] value();’
```

p. 185, code ex. 2: To accommodate the change to the annotation declaration on p. 184, the two instances of `Class<? extends Exception>` in this code fragment must be changed to `Class<? extends Throwable>`.

p. 186, code ex 1: Analogous to the change on p. 184, code ex. 2.

p. 186, par. 3: The prose “you much check” should be replaced by “you must check.”

p. 198, table row 5, col. 3: The constructor invocation “`new TreeMap<K, V>`” is missing parentheses. It should read “`new TreeMap<K, V>()`”

p. 334, Second code example. ***This is a serious error!*** The current code can return null if multiple threads race to initialize the field. Here’s how the code should look:

```
// Double-check idiom for lazy initialization of instance fields
private volatile FieldType field;

private FieldType getField() {
   FieldType result = field;
   if (result != null)    // First check (no locking)
       return result;

   synchronized(this) {
       if (field == null) // Second check (with locking)
           field = computeFieldValue();
       return field;
   }
}
```

To be fixed in fifth printing

p. 20, code ex. 2: The type is missing in the declaration for `INSTANCE`. The declaration should read

```
   public static SpellChecker INSTANCE = new SpellChecker(...);
```

p. 31, par 3: The last sentence listed four class with finalizers that serve as safety nets. The fourth of these classes, java.sql.Connection, is an interface rather than a class and should be removed from the list.

p. 35, code ex. 1: The article `the` is repeated in the comment.

p. 44, par. 1: In the sentence that begins “This is so because most collections, including the `HashSet` used by the `onUnitCircle` method,” “`HashSet`” should be replaced by “set.”

p. 46, code ex. 1: Added “caption” **`// Explicit null check - unnecessary!`** for clarity.

p. 46, code ex. 2: Added “caption” **`// Implicit null check - preferred`** for clarity.

p. 55, par. 1, 4: The string representation `PhoneNumber@163b91`, which occurs twice in paragraph 1 and once in paragraph 4, should be `PhoneNumber@adbbd`, reflecting the actual hash code of the `PhoneNumber` instance for 707-867-5309.

p. 74, par. 5, bullet 2: The sentence that begins “Technically known as *default* access” should begin “Technically known as *package* access.” (The term was changed in the Java 8 edition of the JLS.)

p. 76, par. 1: The fifth sentence begins with a lowercase “a.” It should be capitalized.

p. 94, par. 2: The command line switch `-tag "apiNote:a:API Note:"` should be replaced by the switch `-tag "implSpec:a:Implementation Requirements:"`.

p. 99, par. 2: The interface name `Autocloseable` is incorrectly capitalized. It should be `AutoCloseable`.

p. 102, par 2: The phrase “fields ands methods” in the last sentence should read “fields and methods.”

p. 104, par. 3: The reference to Chapter 6 (for lambdas) should read “Chapter 7.”

p. 108, par. 2: The phrase “fixed of floating point” should be “fixed or floating point.”

p. 114, par. 4: The reference to Chapter 6 (for lambdas) should read “Chapter 7.”

p. 118, code ex. 3: The formatting is wrong, and there are two missing braces. The corrected code looks like this:

```
// Raw iterator type - don't do this!
for (Iterator i = stamps.iterator(); i.hasNext(); ) {
    Stamp stamp = (Stamp) i.next(); // Throws ClassCastException
    stamp.cancel();
}
```

p. 118, code ex. 5: The statement `c.add(new Coin());` should read `stamps.add(new Coin());`

p. 128, par. 2: In the first sentence, the word "use" (before "invoke") is extraneous, and should be removed.

p. 136, par. 4: The word "each" was omitted from the phrase "each time one is requested."

p. 137, par. 4: The word “list” in the phrase “the elements of the list” (after the code example, in the last paragraph on the page) should be replaced by “collection.”

p. 138, par. 2: The word “list” in the phrase “the list is empty” should be replaced by “collection.”

p. 140, par. 1: The phrase “a special kind of parameterized type call a bounded wildcard type” uses the wrong verb tense: “call” should be replaced by “called.”

p. 140, code ex. 3: The type parameter (`Number`) can and should be omitted from the constructor invocation for `Stack` (making use of the diamond operator `<>`, which was added in Java 7).

p. 141, code ex. 3: The article `an` should be replaced by `a` in the phrase `an T producer`, which occurs in the comment.

p. 150, code ex 2: The expression `rnd` should be replaced by `ThreadLocalRandom.current()`.

p. 154, code ex. 1: The expression `type` should be replaced by `Objects.requireNonNull(type)`.

p. 154, par. 4: The reference to Item 30 following the phrase "bounded type parameter" in the last sentence of this paragraph should be a reference to Item 29.

p. 164, par. 2: The item reference for the term “constant variables” should be “(Item 24),” where the term is defined, not “(Item 34),” which is a self-reference.

p. 166, code ex.: I added a parameterless constructor in this edition to reduce the verbosity of this example. This change was misguided, as the whole idea of the pattern is to FORCE the programmer to pick a value for the strategy enum each time a new value is added to the containing enum. Therefore, I eliminated the parameterless `PayrollDay` constructor. To reduce the resulting verbosity, I (implicitly) static-imported the `PayrollDay.PayType` constants. To make this compile, the access level of the constants had to be changed from private to package-private (it is not clear why the compiler enforces this restriction).

p. 181, par. 1: The annotation `@Target(ElementType.METHOD)` was incorrectly rendered as `@Target.get(ElementType.METHOD)`. (The extraneous characters `.get` should be removed.)

p. 191, par. 3: The chapter reference for “Java’s serialization facility” should be “(Chapter 12),” not “(Chapter 6).”

p. 194, par. 4: The Item numbers 14 and 43 were separated by a period rather than a comma.

p. 196, par. 2: The word "method" is incorrectly substituted for the word "class" in the phrase "constant-specific class bodies."

p. 201, par. 1: The name template <Src>`ToObj` was replaced by *Src*`ToObj` (for consistency).

p. 202, par. 5: The interface `java.util.function.Function` should be replaced by the package `java.util.function` (i.e., the extraneous characters `.Function` should be removed).

p. 204, par. 1: The paragraph incorrectly refers to the values of the map `groups` three times. The prose says “list” where it should say “set.”

p.206, par. 1: A sentence reads "This is exactly the same map that was constructed in both previous versions of the program." The word "exactly" in this sentence should be changed to "essentially." In the previous programs, the map values sere sets of strings; in this program they are lists. Since the input dictionary contains no duplicates, the lists won't contain any duplicates either, but the `Set` type better reflects the goal of the program. To make this program produce a `Map<String,Set<String>>`, we would have to use a more complex form of the `groupingBy` collector, lengthening what is already a long item.

p. 211, par. 3: The words "set" and "list" in the third sentence are interchanged. The sentence should read "They return, respectively, a list, a set, and a programmer-specified collection type."

p. 214, par. 4: The phrase “classifier method” should be replaced by “classifier function.”

p. 217, par 2: In the first sentence, the reference to “Item 34” should read “Item 45.”

p. 218, code ex. 1: In the comment `// 2 to the power srcSize`, the identifier `srcSize` should be replaced by `src.size()`.

p.. 225, par. 2: The second occurrence of the class name `ThreadLocalRandom` was not in code font (but should have been). Also, two sentences began with class names (contrary to the style of the book). Fixing these problems caused the page to spill. The paragraph was tweaked to restore the page break.

p. 234, par. 4: The second occurence of the word "that" in the phrase "so that you that don't have to worry" is extraneous and should be removed.

p. 242, par. 2: The method `remove(E)` does not exist; the actual method is `remove(Object)`. The parameter type should be changed in the first sentence, and the second sentence should be simplified to read: “Prior to Java 5 when the `List` interface was “generified,” the corresponding parameter types, `Object` and `int`, were radically different.

p. 252, code ex. 4: There is an extraneous period after the identifier `streamOfOptionals` in the first line of this code example.

p. 257, code ex. 1 and preceding paragraph: The example in the book (`A college degree, such as B.S., M.S. or Ph.D.`) did *not* tickle the undesirable behavior in javadoc, due to changes in the summary sentence parsing code in a release prior to Java 7, but the problem still exists. The broken

example was replaced by `A suspect, such as Colonel Mustard or Mrs. Peacock.`, which does exhibit the undesirable behavior, generating the summary description `A suspect, such as Colonel Mustard or Mrs.`

Incidentally, an even better fix for this problem was provided in Java 10. The summary description in a doc comment can now be indicated explicitly with the `{@summary}` tag:

```
/**
 *  {@summary A suspect, such as Colonel Mustard or Mrs. Peacock}.
 */
public enum Suspect { ... }
```

This solution will *not* be described in the third edition of *Effective Java*, which only covers releases up to Java 9.

p. 258, code ex. 1: The closing brace in **`{@index IEEE 754}`** should have been boldfaced like the rest of the tag.

p. 284, par. 3: The phrase "the program with throw" should read "the program will throw."

p. 341, par. 2: The word “to” is missing from the phrase “what can you do to defend .”

P. 370: Two item numbers were duplicated in the second column of this table (“81, 81,” should be “81,” and “86, 86,” should be “86,”).

[iv, 31, 259, 276, 280, 380, 391]