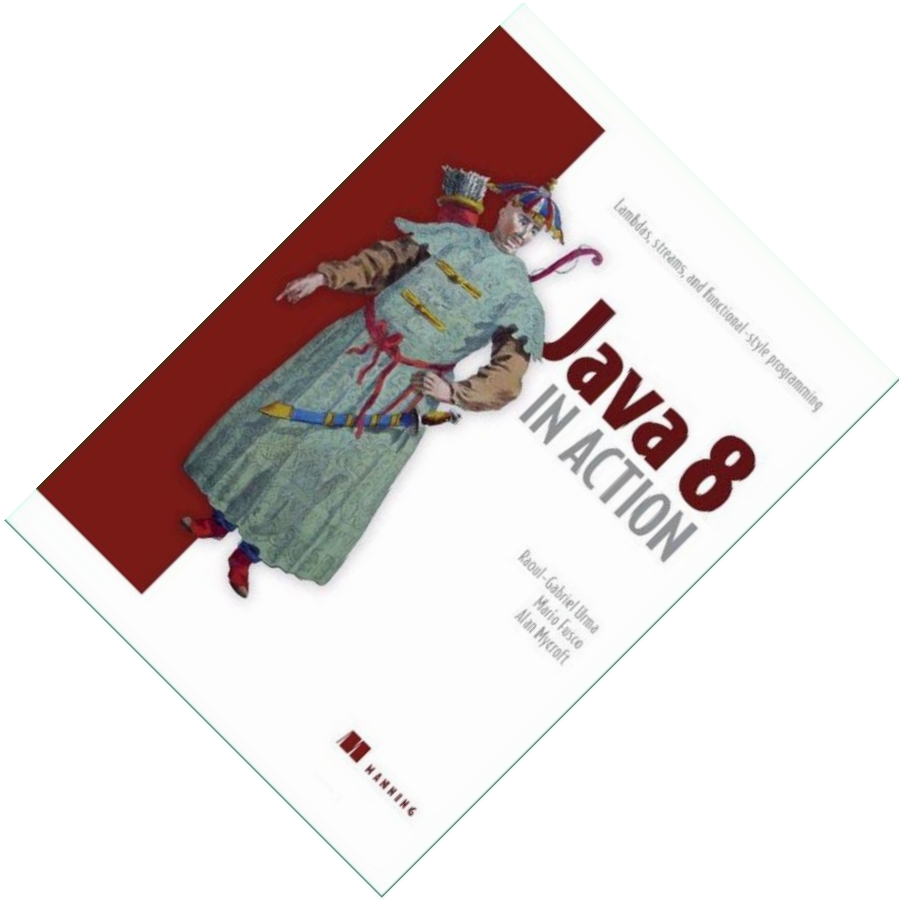
Java 8 quizzes

2014 Olivier Dupuy

with some help from



Java 8: Q 1/8 Compact code thanks to the streams API

```
// Q1: what does it do?
Arrays.stream(new File("c:/").listFiles(File::isHidden)).forEach(
        System.out::println);

// Q2: what does it do?
LongSummaryStatistics stats = Arrays.stream(
        new File("c:/").listFiles(File::isHidden)).collect(
        Collectors.summarizingLong(File::length));
System.out.println(stats);

// Q3: what does it do?
System.out.println(Arrays
        .stream(new File("c:/").listFiles()).filter(File::isHidden)
        .sorted(Comparator.comparingLong(File::length).reversed())
        .limit(5).sorted()
        .peek(t -> System.out.println(t.getName() + ' ' + t.length()))
        .collect(Collectors.summarizingLong(File::length)));
```

Java 8: Q 2/8 A lambda is some kind of anonymous class

```
3  public class Java8Quiz2ContentTest {
4      Runnable myRunnable = new Runnable() {
5          public void run() { System.out.println("in runnable 0");     }
6      };
7      @SuppressWarnings("unused")
8      public Java8Quiz2ContentTest() {
9          class MyLocalClass extends Thread {
10             public void run() { System.out.println("in runnable 1"); }
11         };
12         Thread thread2 = new Thread(() -> System.out.println("in runnable 2"));
13     }
14     static class MyStaticClass {
15         Thread thread4 = new Thread(() -> System.out.println("in runnable 3"));
16     }
17     class MyInnerClass implements Runnable {
18         public void run() { System.out.println("in runnable 4"); }
19     }}
```

- Q: Give the names of all the class files in the folder target/classes/org/java8…

Java 8: Q 3/8 Stack trace with a lambda

```
4  import java.util.Objects;
5
6  public class Java8Quiz3StackTraceInLambda {
7
8      public static void main(String[] args) {
9          Apple appleII = new Apple("Special", 2) { // name and weight
10             @Override
11             public int getWeight() {
12                 new RuntimeException().printStackTrace();
13                 return super.getWeight();
14             };
15         };
16         Apple appleIII = new Apple("Apple", 3);
17         Comparator<Apple> byWeight = (a, b) -> a.getWeight() - b.getWeight();
18         Objects.compare(appleII, appleIII, byWeight);
19
20         // what will show the stack trace for this comparison?
```

Java 8: Q 4/8 Matching the method

- Given **public class** File {
  … **public** File[] listFiles(FileFilter filter) {…}; …}

  and @FunctionalInterface **interface** FileFilter {
  boolean accept(File pathname); }

  and

```
static class FileUtils {
    static boolean isHidden(File file) { return file.isHidden(); }
    boolean isHiddenFile(File file)    { return file.isHidden(); } }
    Predicate<File> hiddenFilePredicatePlusPlus = new Predicate<File>() {
        public boolean test(File t) { return t.isHidden(); };
        public void otherMethod() {} };
    Function<File,Boolean> myFunction = new Function<File,Boolean>() {
        @Override public Boolean apply(File t) { return t.isHidden(); } };
```

- **Which lines compile properly retrieving the hidden files?**

1. **new** File("...").listFiles(File::isHidden);
2. **new** File("...").listFiles(t -> t.isHidden());
3. **new** File("...").listFiles(myFunction);
4. **new** File("...").listFiles(myFunction::apply);
5. **new** File("...").listFiles(FileUtils::isHidden);
6. **new** File("...").listFiles(t -> FileUtils::isHidden);
7. **new** File("...").listFiles(t -> FileUtils::isHidden(t));
8. **new** File("...").listFiles(t -> FileUtils::isHiddenFile);
9. **new** File("...").listFiles(hiddenFilePredicatePlusPlus::test);

Java 8: Q 5/8 Lambda in a loop

- Given:

```
5  public class LambadInTheLoop {
6      static Integer myStatic = Integer.valueOf(1);
7      final Integer myMember = Integer.valueOf(2);
8
9      public static void main(String[] args) {
10         LambadInTheLoop instance = new LambadInTheLoop();
11         for (int i = 0; i < 2; i++) {
12             final int j = 2 * i;
13             final int k = 4;
14
15             System.out.println((Supplier<String>) () -> {
16                 return "aaa[" + j + instance.myMember + "]"; });
17
18             System.out.println((Supplier<String>) () -> {
19                 return "bbb[" + System.getenv("yes") + " " + myStatic + k + "]"; });
20             System.out.println();
21         } } }
```

Explain the following output:

```
org.java8.Java8Quiz5LambadInTheLoop$$Lambda$1/12251916@d46ca6
org.java8.Java8Quiz5LambadInTheLoop$$Lambda$2/18340259@105068a

org.java8.Java8Quiz5LambadInTheLoop$$Lambda$1/12251916@132e575
org.java8.Java8Quiz5LambadInTheLoop$$Lambda$2/18340259@105068a
```

Java 8: Q 6/8 Create your own collector

Given this code using the default LongSummaryStatistics collector on the end result of the stream

```
19 public class Java8Quiz6MyCollector {
20     public static void main(String[] args) {
21         long[] myValues = new long[] { 1, 3, 5, 7, 8, 9, 10, 11, 12, 13, 14,
22                 14, 14, 14, 15, 16, 17, 18 };
23
24         LongSummaryStatistics stats = Arrays.stream(myValues).boxed()
25                 .collect(Collectors.summarizingLong(Long::longValue));
26         System.out.println(stats);
27         // LongSummaryStatistics{count=18, sum=201, min=1, average=11.166667, max=18}
```

and this collector factory (static method from the Collector interface)

```
243     /**
244      * Returns a new {@code Collector} described by the given {@code supplier},
245      * {@code accumulator}, and {@code combiner} functions.  The resulting
246      * {@code Collector} has the {@code Collector.Characteristics.IDENTITY_FINISH}
247      * characteristic.
248      *
249      * @param supplier The supplier function for the new collector
250      * @param accumulator The accumulator function for the new collector
251      * @param combiner The combiner function for the new collector
252      * @param characteristics The collector characteristics for the new
253      *                        collector
254      * @param <T> The type of input elements for the new collector
255      * @param <R> The type of intermediate accumulation result, and final result,
256      *           for the new collector
257      * @throws NullPointerException if any argument is null
258      * @return the new {@code Collector}
259      */
260     public static<T, R> Collector<T, R, R> of(Supplier<R> supplier,
261                                               BiConsumer<R, T> accumulator,
262                                               BinaryOperator<R> combiner,
263                                               Characteristics... characteristics) {
```

how can you create your own collector with the same results as LongSummaryStatistics but giving the standard deviation as well?

Java 8: Q 7/8 Changes to the Java interface

Default method

- Q1 How can I create a default method?
- Q2 Why creating a default method?
- Q3 Any restriction for default methods?

Static method

- Q4 How can I create a static method?
- Q5 Why creating a static method?
- Q6 Any restriction for static methods?

Functional method and functional interface

- Q7 How can I create a functional method?
- Q8 Why creating a functional method?
- Q9 Any restriction for functional methods?
- Q10 What are the benefits of functional interfaces?

Java 8: Q 8/8 New FunctionalInterface (FI) interfaces

- What can I do with the new FIs?

- Q1 Supplier<T>
- Q2 Consumer<T> and BiConsumer<T,U>
- Q3 Predicate<T>
- Q4 Function<T,R> and BiFunction<T,U,R>
- Q5 UnaryOperator<T> and BinaryOperator<T>

- Q6 What should I do if I pass or return primitives?

Java 8: Have more fun

- Mother's and Father's Day are coming. Put *Java 8 in action, Manning 2014* in your basket.
- As a certified OCP 7, for a limited time pass the beta exam for the upgrade to Java 8 programmer for 50 USD vs. 250 usually.
- Last public Java 7 release in April !
- Be like the Chinese mandarin of war at right. Get ready for the real action !

.