Preparing for the Java cert and learning new features (part 2 - Java 12 to 17)

Jeanne Boyarsky & Scott Selikoff

Wednesday, April 13, 2022

DevNexus

speakerdeck.com/boyarsky

@JeanneBoyarsky @ScottSelikoff

About Us

- Java Developer
- CodeRanch Mod
- JUG Leader
- Java Champion

@JeanneBoyarsky

- Java Developer
- CodeRanch Mod
- JUG Leader
- Software Engineer

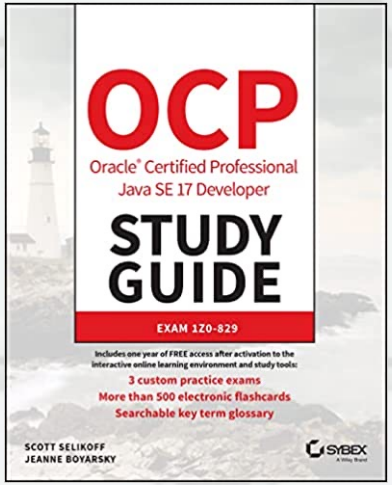
@ScottSelikoff

Jeanne & Scott's Java 8/11 Cert Books

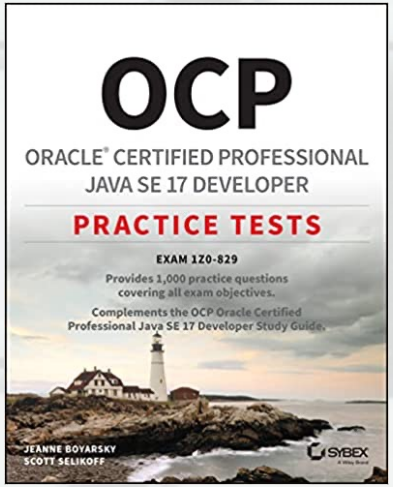
- OCA: Java 8 Programmer I Study Guide
- OCP: Java 8 Programmer II Study Guide
- OCA / OCP Java 8 Practice Tests
- OCP Java 11 Programmer I Study Guide
- OCP Java 11 Programmer II Study Guide
- OCP Java 11 Developer Complete Study Guide
- OCP Java 11 Practice Tests

Win a book at the end!

Pre-order Java 17 Cert Books



May 2022



Sept 2022

@JeanneBoyarsky @ScottSelikoff

Available exams

	Java 8	Java 11		Java 17
Associate	808			
Professional	809	~~815 + 816~~	819	829

Also 1Z0-811 (Foundations)

Agenda

- Text blocks
- Switch Expressions & Pattern Matching
- Records & Sealed classes

- Also on the exam – other new features like compact number formatting and helpful null pointers

Text Blocks

What's wrong?

```
String old = "devnexus,Atlanta,"session,workshop"
        + "meetup,Various,lecture\n";
```

Doesn't compile: missing \

Missing quote on line 1

Missing line break

Compare

```
String old =
        "devnexus,Atlanta ,\"session,workshop\"\n"
        + “meetup,Various,lecture\n";

String textBlock = """
        devnexus,Atlanta,"session,workshop"
        meetup,Various,lecture
        """;
```

Easier to read and code!

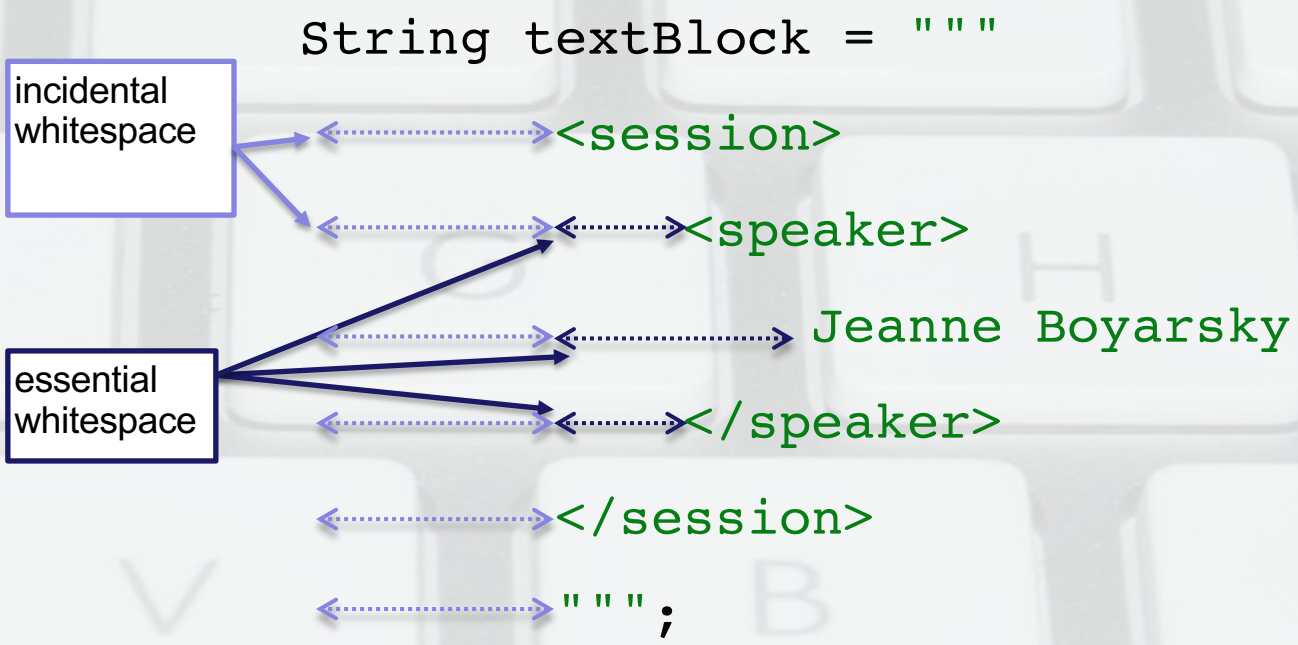
Text Block Syntax

start block

```
String textBlock = """
        devnexus,Atlanta,"session,workshop"
        meetup,Various,lecture
        """;
```

end block

Essential Whitespace



Ending lines

```
String textBlock = """
        <session>
            <speaker>
                Jeanne Boyarsky     \s
            </speaker>
            <title>
                Becoming one of the first Java 17 \
                certified programmers \
                (and learning new features)
            </title>
        </session>
        """;
```

new escape character
keeps trailing whitespace

tab

continue on next line
without a line break

New lines

```
String textBlock = """
        <session>\n
            <speaker>
                Jeanne\nBoyarsky
            </speaker>
        </session>""";
```

Two new lines (explicit and implicit)

One new line (explicit)

no line break at end

Escaping Three Quotes

```
String textBlock = """
        better \"""
        but can do \"\"\"
        """;
```

Indent

```
String option1 = "    a\n    b\n    c\n";
String option2 = "a\nb\nc\n".indent(3);
String option3 = """
        a
        b
        c
        """.indent(3);
```

Which do you like best?

Also normalizes (bye \r)

Strip Indent

Method	From beginning	From end	Per line
strip()	Leading	Trailing	No
stripIndent()	Incidental	Incidental	Yes
stripLeading()	Leading	n/a	No
stripTrailing()	n/a	Trailing	No

Question 1

How many lines does this print out?

```
String sql = """
        select * \n
        from mytable;
        """;
System.out.print(sql);
```

A. 2

B. 3

C. 4

D. Does not compile

Question 1

How many lines does this print out?

```
String sql = """
        select
        from m
        """;
System.out.print(sql);
```

C

A. 2

B. 3

C. 4

D. Does not compile

Question 2

How many lines does this print out?

```
String sql = """
        select * \
        from mytable;""".stripIndent();
System.out.print(sql);
```

A. 1

B. 2

C. 3

D. Does not compile

Question 2

How many lines does this print out?

```
String sql = """
        select
        from
mytable;""".stripIndent();
System.out.print(sql);
```

A

A. 1

B. 2

C. 3

D. Does not compile

Question 3

How many lines does this print out?

```
String sql = """
        select *     \s
        from mytable;
    ".stripIndent();
System.out.print(sql);
```

A. 1

B. 2

C. 3

D. Does not compile

Question 3

How many lines does this print out?

```
String sql = """
        select
        from m
    ".stripIndent();
System.out.print(sql);
```

D

A. 1

B. 2

C. 3

D. Does not compile

Question 4

How many whitespace characters are removed by strip()?

```
String sql = """
            select "name"
            from mytable;
        """;
```

A. 1

B. 2

C. 3

D. Does not compile

Question 4

How many whitespace characters are removed by strip()?

```
String sql = "
        sele
        from mytable;
    """;
```

C

A. 1

B. 2

C. 3

D. Does not compile

Question 5

How many escapes can be removed without changing the behavior?

```
String sql = """
             select \"name\"
             from mytable \
             where value = '\"\"'''
           """;
```

A. 2

B. 3

C. 4

D. Does not compile

Question 5

How many escapes can be removed without changing the behavior?

```
String sql = """
        select \"
        from mytable \
        where value = '\"\"'''
    """;
```

B

A. 2

B. 3

C. 4

D. Does not compile

Switch Expressions and Pattern Matching

What does this print?

```
String store = "Ponce City";

switch(store) {
    case "Ponce City" : System.out.println("GA");
    case "Crayola" : System.out.println("PA");
    default: System.out.println("anywhere");
}
```

Three lines (missing break)

Switch Expressions

```
String store = "Ponce City";

switch(store) {
    case "Ponce City" -> System.out.println("GA");
    case "Crayola", "H&R"
                    -> System.out.println("PA");
    default -> System.out.println("anywhere");
}
```

Arrow labels

No break keyword

Multiple values

Switch Expressions

```
String store = "Ponce City";

String output = switch(store) {
    case "Ponce City" -> "GA";
    case "Crayola" -> "PA";
    default -> "anywhere";
};
System.out.println(output);
```

Switch returns values

More features

```
String output = switch(store) {
    case "Ponce City" -> "GA";
    case "Legoland" -> {
        int random = new Random().nextInt();
        String city = random % 2 == 0
            ? "GA" : "Carlsbad";
        yield city;
    }
    default -> throw new
        IllegalArgumentException("Unknown");
};
System.out.println(output);
```

Block

yield

throws exception
so no return value
needed

Is this legal?

```
String store = "Legoland";
String output = switch(store) {
   case "NYC" -> 0;
   case "Legoland" -> {
      yield "Carlsbad";
   }
   default -> "PA";
};
System.out.println(output);
```

No, return types must be compatible

Yield with Switch Stmt

```
enum Position { TOP, BOTTOM };

Position pos = Position.TOP;

int stmt = switch(pos) {
    case TOP: yield 1;
    case BOTTOM: yield 0;
};

int expr = switch(pos) {
    case TOP -> 1;
    case BOTTOM -> 0;
};
```

Same!

Missing value

```
enum Position { TOP, BOTTOM };

Position pos = Position.TOP;

int stmt = switch(pos) {
    case TOP: yield 1;
};

int expr = switch(pos) {
    case BOTTOM -> 0;
};
```

Does not compile because assigning value and not all values covered

Missing values

```
int greeting = 10;

String output = switch(greeting) {
    case 1 -> "Welcome!";
    case 10 -> "Goodbye";
};

System.out.println(output);
```

Does not compile.
Requires default branch

Pattern matching for if

```
if (num instanceof Integer) {
    Integer numAsInt = (Integer) num;
    System.out.println(numAsInt);
}
if (num instanceof Double) {
    Double numAsDouble = (Double) num;
    System.out.println(numAsDouble.intValue());
}

if (num instanceof Integer numAsInt) {
    System.out.println(numAsInt);
}
if (num instanceof Double numAsDouble) {
    System.out.println(numAsDouble.intValue());
}
```

Pattern variable

Flow Scope

```
if (num instanceof Double d1
    && d1.intValue() % 2 == 0) {

    System.out.println(d1.intValue());
}

if (num instanceof Double d2
    || d2.intValue() % 2 == 0) {

    System.out.println(d2.intValue());
}
```

Compiles

Does not compile because d2 might not be double

Does this compile?

```
if (num instanceof Double n)
    System.out.println(n.intValue());

if (num instanceof Integer n)
    System.out.println(n);
```

Yes. Only in scope for if statement

Does this compile?

```
if (num instanceof Double n)
    System.out.println(n.intValue());

System.out.println(n.intValue());
```

No. If statement is over

Does this compile?

```
if (!(num instanceof Double n)) {
    return;
}
System.out.println(n.intValue());
```

Yes. Returns early so rest is like an else

Does this compile?

```
if (!(num instanceof Double n)) {
    return;
}
System.out.println(n.intValue());

if (num instanceof Double n)
    System.out.println(n.intValue());
```

No. n is still in scope

Reusing a variable

```
if (num instanceof Integer numAsInt) {
    numAsInt = 6;
    System.out.println(numAsInt);
}
```

Legal. please don't.

Question 6

What is output?

```
char ch = 'b';
int count = 0;
switch (ch) {
    case 'a' -> count++;
    case 'b' -> count+=2;
    case 'c' -> count+=3;
 }
 System.out.println(count);
```

A. 1
B. 2
C. 5
D. Does not compile

Question 6

What is output?

```
char ch = 'b';
int count = 0;
switch (ch) {
    case 'a' -> count++;
    case 'b' -> count+=2;
    case 'c' -> count+=3;
 }
 System.out.println(count);
```

B

A. 1
B. 2
C. 5
D. Does not compile

Question 7

What can fill in the blank to have the code print 2?

```
char ch = 'b';

_______ value = switch (ch) {
    case 'a' -> 1;
    case 'b' -> 2L;
    case 'c' -> 3.0;
    default -> 4;
};
System.out.println(value);
```

A. int
B. Object
C. Either A or B
D. None of the above

Question 7

What can fill in the blank to have the code print 2?

```
char ch = 'b';

_______ value = {
    case 'a' -> 1;
    case 'b' -> 2L;
    case 'c' -> 3.0;
    default -> 4;
};
System.out.println(value);
```

B

A. int
B. Object
C. Either A or B
D. None of the above

Question 8

What does the following print?

```
Object robot = "694";

if (robot instanceof String s) {
    System.out.print("x");
}
if (robot instanceof Integer s) {
    System.out.print("y");
}
System.out.println(robot);
```

A. x694
B. xy694
C. y694
D. Does not compile

Question 8

What does the following print?

```
Object robot = "694";

if (robot instanceof String s) {
    System.out.print("x");
}
if (robot instanceof Integer s) {
    System.out.print("y");
}
System.out.println(robot);
```

A. x694
B. xy694
C. y694
D. Does not compile

A

Question 9

Which lines have s in scope?

```
Object robot = "694";

if (robot instanceof String s) {
    // line 1
}
if (robot instanceof int i) {
    // line 2
}
// line 3
```

A. 1
B. 1 and 3
C. 1, 2 and 3
D. Does not compile

Question 9

Which lines have s in scope?

```
Object robot = "694";

if (robot instanceof String s) {
    // line 1
}
if (robot instanceof int i) {
    // line 2
}
// line 3
```

A. 1
B. 1 and 3
C. 1, 2 and 3
D. Does not compile

D

Question 10

What is true about this class?

```
class Sword {
    int length;

    public boolean equals(Object o) {
        if (o instanceof Sword sword)
            return length == sword.length;
        return false;
    } }
 // assume hashCode properly implemented
```

A. equals() is correct
B. equals() is incorrect
C. equals() does not compile

Question 10

What is true about this class?

```
class Sword {
    int length;

    public boolean equals(Object o) {
        if (o instanceof Sword sword)
            return length == sword.length;
        return false;
    } }
// assume hashCode properly implemented
```

A. equals() is correct
B. equals() is incorrect
C. equals() does not compile

A

@JeanneBoyarsky @ScottSelikoff

Records and Sealed Classes

Immutable class

1. Make fields final and private
2. Don’t provide setters
3. No subclasses (ex: make class final)
4. Write constructor taking all fields

POJO

- constructor
- toString()
- hashCode() - more rules
- equals() - still more rules

Simple Record

```
public record Book (String title, int numPages) {
}
```

New type

Automatically get
* final record
* private final instance variables
* public accessors/getters
* constructor taking both fields
* equals(), hashCode() and toString()

Using the Record

```
Book book = new Book("Breaking and entering", 289);

System.out.println(book.title());
System.out.println(book.toString());
```

No “get”

Outputs:
Breaking and entering
Book[title=Breaking and entering, numPages=289]

Add/change methods

```
public record Book (String title, int numPages) {

    @Override
    public String title() {
        return '"' + title + '"';
    }

    public boolean isLong() {
        return numPages > 300;
    }

}
```

Change behavior

Custom method

Immutability

```
public record Book (String title, int numPages,
   List<String> chapters) {
}


Book book = new Book("Breaking and entering", 289,
   chapters);

chapters.add("2");
book.chapters().add("3");
System.out.println(book.chapters());
```

Prints [1,2,3] because shallow immutability

Now immutable

```
public record Book (String title, int numPages,
   List<String> chapters) {

    public Book {
       chapters = List.copyOf(chapters);
    }
}
```

Compact constructor

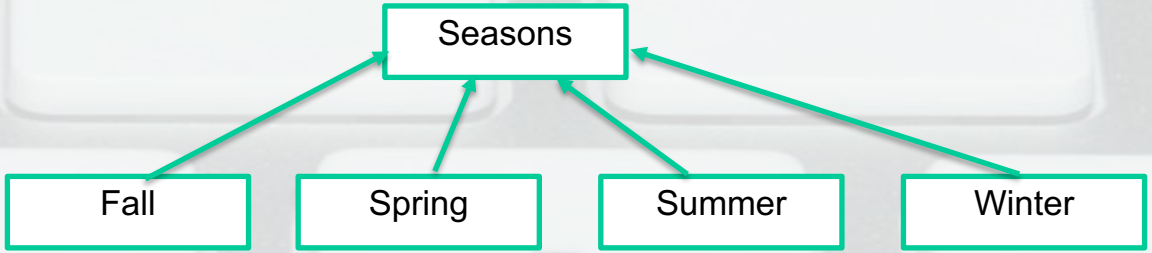
Make immutable copy

Must match record access modifier

Sealed classes

```
public abstract sealed class Seasons
   permits Fall, Spring, Summer, Winter { }

final class Fall extends Seasons {}
final class Spring extends Seasons {}
final class Summer extends Seasons {}
final class Winter extends Seasons {}
```



Subclass modifiers

Modifier	Meaning
`final`	Hierarchy ends here
`non-sealed`	Others can subclass
`sealed`	Another layer

Sealed interface

```
public sealed interface TimeOfDay
   permits Morning, Hour, Evening {
    boolean early();
}
public non-sealed class Morning implements TimeOfDay {
    public boolean early() { return true; }
}
public non-sealed interface Hour extends TimeOfDay {}
public record Evening(int hour) implements TimeOfDay {
    public boolean early() { return false; }
}
```

Records are implicitly final

Does this compile?

```
public abstract sealed class Seasons { }

final class Fall extends Seasons {}
final class Spring extends Seasons {}
final class Summer extends Seasons {}
final class Winter extends Seasons {}
```

Yes, but only when they are in the same file.

Question 11

How many lines need to be removed for this code to compile?

```
public record BBQ(String type) {}
public static void main(String[] args) {
 BBQ bbq = new BBQ("chicken");
 System.out.println(bbq.setType("pork"));
 System.out.println(bbq.getType());
 System.out.println(bbq.equals(bbq));
}
```

A. 0
B. 1
C. 2
D. None of the above

Question 11

How many lines need to be removed for this code to compile?

```
public record BBQ(String type) {}
public static v                tring[] args) {
 BBQ bbq = new                 en");
 System.out.println(bbq.setType("pork"));
 System.out.println(bbq.getType());
 System.out.println(bbq.equals(bbq));
}
```

C

A. 0
B. 1
C. 2
D. None of the above

Question 12

What does this output?

```
public record BBQ(String type) {
    BBQ {
        type = type.toUpperCase();
    } }
public static void main(String[] args) {
    BBQ bbq = new BBQ("chicken");
    System.out.println(bbq.type());
}
```

A. chicken
B. CHICKEN
C. Does not compile
D. None of the above

Question 12

What does this output?

```
public record BBQ(String type) {
    BBQ {
        type =                    erCase();
    } }
public static void main(String[] args) {
    BBQ bbq = new BBQ("chicken");
    System.out.println(bbq.type());
}
```

C

A. chicken
B. CHICKEN
C. Does not compile
D. None of the above

Question 13

How many compiler errors are in the following code?

```
public final record BBQ(String type) {
  { type = ""; }
  public BBQ(String type) {
      type = type.toUpperCase();
  }
  public void type() { return ""; }
  public String toString() { return ""; }
}
```

A. 1
B. 2
C. 3
D. 4

Question 13

How many compiler errors are in the following code?

```
public final record BBQ(String type) {
  { type = ""; }
  public BBQ(St                    {
      type = typ                   case();
  }
  public void type() { return ""; }
  public String toString() { return ""; }
}
```

C

A. 1
B. 2
C. 3
D. 4

Question 14

What does this output?

```
public record BBQ(String type)
    implements Comparable<BBQ> {
    public int compareTo(BBQ bbq) {
        return type.compareTo(bbq.type);
    } }
public static void main(String[] args) {
    BBQ beef = new BBQ("beef");
    BBQ pork = new BBQ("pork");
    System.out.println(pork.compareTo(beef));
}
```

A. Negative #
B. Positive #
C. 0
D. Does not compile

Question 14

What does this output?

```
public record BBQ(String type)
    implements Comparable<BBQ> {
    public int compareTo(BBQ bbq) {
        return type.compareTo(bbq.type);
    } }
public static void main(String[] args) {
    BBQ beef = new BBQ("beef");
    BBQ pork = new BBQ("pork");
    System.out.println(pork.compareTo(beef));
}
```

B

A. Negative #
B. Positive #
C. 0
D. Does not compile

@JeanneBoyarsky @ScottSelikoff

Question 15

How many compiler errors are in this code?

```
public sealed class Phone {
    class IPhone extends Phone {
    }
    class Android extends Phone {
    }
}
```

A. 0
B. 1
C. 2
D. 3

Question 15

How many compiler errors are in this code?

```
public sealed class Phone {
    class IPhone extends Phone {
    }
    class Android extends Phone {
    }
}
```

C

A. 0
B. 1
C. 2
D. 3

Book Giveaway

@JeanneBoyarsky @ScottSelikoff

Question 16

Which is true about this text block?

```
String sql = """select *
        from mytable \
        where weather = 'snow';
        """;
```

A. Has incidental whitespace

B. Has essential whitespace

C. Both A and B

D. Doesn't compile

Question 16

Which is true about this text block?

```
String sql = """select *
        from m
        where                    'snow';
        """;
```

D

A. Has incidental whitespace

B. Has essential whitespace

C. Both A and B

D. Doesn't compile

Question 17

How many changes are needed to have this code print 2?

```
char ch = 'b';
int value = switch (ch) {
    case 'a' -> 1;
    case 'b' -> yield 2;
    case 'c' -> 3;
 }
 System.out.println(value);
```

A. 1
B. 2
C. 3
D. 4

Question 17

How many changes are needed to have this code print 2?

```
char ch = 'b';
int value = swi
    case 'a' ->
    case 'b' -> yield 2;
    case 'c' -> 3;
 }
 System.out.println(value);
```

C

A. 1
B. 2
C. 3
D. 4

Question 18

What does printLength(3) print?

```
class Sword {
    int length = 8;
    public void printLength(Object x) {
        if (x instanceof Integer length)
            length = 2;
        System.out.println(length);
    } }
```

A. 2
B. 3
C. 8
D. Does not compile

Question 18

What does printLength(3) print?

```
class Sword {
    int length = 8;
    public void [illegible]h(Object x) {
        if (x in[illegible]teger length)
            length = 2;
        System.out.println(length);
    } }
```

C

A. 2
B. 3
C. 8
D. Does not compile

Question 19

What does this output?

```
record BBQ(String type) {
    BBQ {
        type = type.toUpperCase();
    } }
public static void main(String[] args) {
    BBQ bbq = new BBQ("chicken");
    System.out.println(bbq.type());
}
```

A. chicken
B. CHICKEN
C. Does not compile
D. None of the above

Question 19

What does this output?

```
record BBQ(String type) {
    BBQ {
        type = [illegible]erCase();
    } }
public static void main(String[] args) {
    BBQ bbq = new BBQ("chicken");
    System.out.println(bbq.type());
}
```

B

A. chicken
B. CHICKEN
C. Does not compile
D. None of the above