Exam Topics: Java SE 17 Developer

Please note that all information presented in this appendix is valid as of November 2022. It is imperative that readers visit the exam website mentioned in this appendix regularly while preparing for the exam, as Oracle is known to change the exam objectives intermittently.

Exam Name: *Java SE 17 Developer*	Duration: *90 minutes*
Exam Number: *1Z0-829*	Number of Questions: *50*
Certification: *Oracle Certified*	Passing Score: 68%
Professional: Java SE 17 Developer	Exam Format: *Multiple Choice*
	Exam Price: *$245*

Candidates must pass the *Java SE 17 Developer* exam in order to qualify as an *Oracle Certified Professional: Java SE 17 Developer*. Pertinent information about this exam can be found at:

Click here to view code image

```
https://education.oracle.com/java-se-17-developer/pexam_1Z0-829
```

The web page also lists the exam topics defined by Oracle. The topics are organized in *sections*, and each section is *reproduced verbatim* in this appendix. For each section, we have provided references to where in the book the exam topics (we call them *objectives*) in the section are covered. In addition, the extensive index at the end of the book can also be used to look up specific topics.

General information about taking the exam can be found in **Appendix A**, **p. 1615**. Oracle has also specified certain important assumptions about the exam questions, which can also be found in **Appendix A**, **p. 1619**. In addition to the exam objectives and the assumptions, the exam web page lists a number of topics that the candidate is expected to know, but the exam will not include any direct questions on these topics. We have classified them as *supplementary topics* (p. 1627).

Section 1: Handling date, time, text, numeric and boolean values	Chapters: 2, 8, 17
[1.1] Use primitives and wrapper classes including Math API, parentheses, type promotion, and casting to evaluate arithmetic and boolean expressions	§2.2, p. 41 *to* §2.19, p. 92 §8.3, p. 429 §8.6, p. 478
[1.2] Manipulate text, including text blocks, using String and StringBuilder classes	§8.4, p. 439 §8.5, p. 464
[1.3] Manipulate date, time, duration, period, instant and time-zone objects using Date-Time API	§17.1, p. 1024 *to* §17.7, p. 1072
Section 2: Controlling Program Flow	Chapter 4
[2.1] Create program flow control constructs including if/else, switch statements and expressions, loops, and break and continue statements	§4.1, p. 152 *to* §4.13, p. 184
Section 3: Utilizing Java Object-Oriented Approach	Chapters: 3, 5, 6, 9, 10, 14
[3.1] Declare and instantiate Java objects including nested class objects, and explain the object life-cycle including creation, reassigning references, and garbage collection	§9.1, p. 491 *to* §9.6, p. 521 §10.1, p. 533 *to* §10.4, p. 537
[3.2] Create classes and records, and define and use instance and static fields and methods, constructors, and instance and static initializers	§3.1, p. 99 *to* §3.8, p. 112 §5.14, p. 299 §10.5, p. 540 *to* §10.9, p. 555

Section 1: Handling date, time, text, numeric and boolean values	Chapters: 2, 8, 17
[3.3] Implement overloading, including var-arg methods	§3.6, p. 108 §3.11, p. 136 §5.1, p. 202
[3.4] Understand variable scopes, use local variable type inference, apply encapsulation, and make objects immutable	§3.13, p. 142 §6.1, p. 324 §6.6, p. 352 §6.7, p. 356
[3.5] Implement inheritance, including abstract and sealed classes. Override methods, including that of Object class. Implement polymorphism and differentiate object type versus reference type. Perform type casting, identify object types using instanceof operator and pattern matching	§5.1, p. 191 *to* §5.4, p. 218 §5.11, p. 269 §5.12, p. 278 §5.15, p. 311 §14.1, p. 743 *to* §14.3, p. 753
[3.6] Create and use interfaces, identify functional interfaces, and utilize private, static, and default interface methods	§5.6, p. 237
[3.7] Create and use enumerations with fields, methods and constructors	§5.13, p. 287
Section 4: Handling Exceptions	Chapter 7
[4.1] Handle exceptions using try/catch/finally, try-with-resources, and multi-catch blocks, including custom exceptions	§7.2, p. 375 §7.3, p. 375 §7.6, p. 397 §7.7, p. 407
Section 5: Working with Arrays and Collections	Chapters: 3, 11, 12, 14, 15
[5.1] Create Java arrays, List, Set, Map and Deque collections, and add, remove, update, retrieve and sort their elements	§3.9, p. 117 §11.1, p. 565 *to* §11.13, p. 623 §12.1, p. 644 *to* §12.8, p. 662 §14.4, p. 761

Section 1: Handling date, time, text, numeric and boolean values	Chapters: 2, 8, 17
	§14.5, p. 769 §15.1, p. 783 *to* §15.12, p. 864
Section 6: Working with Streams and Lambda expressions	Chapters: 13, 16
[6.1] Use Java object and primitive Streams, including lambda expressions implementing functional interfaces, to supply, filter, map, consume, and sort data	§13.1, p. 675 *to* §13.14, p. 733 §16.3, p. 884 *to* §16.7, p. 946
[6.2] Perform decomposition, concatenation and reduction, and grouping and partitioning on sequential and parallel streams	§16.7, p. 946 §16.8, p. 978 §16.9, p. 1009
Section 7: Package and deploy Java code and use the Java Platform Module System	Chapter 19
[7.1] Define modules and their dependencies, expose module content including for reflection. Define services, producers, and consumers	§19.1, p. 1163 *to* §19.5, p. 1179 §19.8, p. 1191 §19.9, p. 1196
[7.2] Compile Java code, produce modular and non-modular jars, runtime images, and implement migration using unnamed and automatic modules	§19.6, p. 1186 §19.7, p. 1189 §19.10, p. 1204 *to* §19.14, p. 1218
Section 8: Manage concurrent code execution	Chapters: 16, 22, 23
[8.1] Create worker threads using Runnable and Callable, manage the thread lifecycle, including automations provided by different Executor services and concurrent API	§22.1, p. 1367 *to* §22.5, p. 1408 §23.2, p. 1423

Section 1: Handling date, time, text, numeric and boolean values	**Chapters: 2, 8, 17**
[8.2] Develop thread-safe code, using different locking mechanisms and concurrent API	§23.4, p. 1451 *to* §23.7, p. 1482
[8.3] Process Java collections concurrently including the use of parallel streams	§16.9, p. 1009 §23.7, p. 1482
Section 9: Use Java I/O API	**Chapters: 20, 21**
[9.1] Read and write console and file data using I/O Streams	§20.1, p. 1233 *to* §20.4, p. 1256
[9.2] Serialize and de-serialize Java objects	§20.5, p. 1261
[9.3] Create, traverse, read, and write Path objects and their properties using java.nio.file API	§21.1, p. 1287 *to* §21.8, p. 1345
Section 10: Access databases using JDBC	**Chapter 24**
[10.1] Create connections, create and execute basic, prepared and callable statements, process query results and control transactions using JDBC API	§24.1, p. 1512 *to* §24.8, p. 1545
Section 11: Implement Localization	**Chapter 18**
[11.1] Implement localization using locales, resource bundles, parse and format messages, dates, times, and numbers including currency and percentage values	§18.1, p. 1096 *to* §18.7, p. 1139
Section 12: Supplementary Topics	**Chapters: 11, 25**
[12.1] Understand the basics of Java Logging API.	*See* Appendix G, p. 1747.

Section 1: Handling date, time, text, numeric and boolean values	**Chapters: 2, 8, 17**
[12.2] Use Annotations such as Override, FunctionalInterface, Deprecated, SuppressWarnings, and SafeVarargs.	*Standard annotations are covered in* **§25.5**, **p. 1577***, and are used throughout the book. See also the index.*
[12.3] Use generics, including wildcards.	*Generics are covered in* **Chapter 11**, **p. 563**.