11 Exception Handling: A Deeper Look

Objectives

In this chapter you'll:

- Learn why exception handling is an effective mechanism for responding to runtime problems.
- Use `try` blocks to delimit code in which exceptions might occur.
- Use `throw` to indicate a problem.
- Use `catch` blocks to specify exception handlers.
- Learn when to use exception handling.
- Understand the exception class hierarchy.
- Use the `finally` block to release resources.
- Chain exceptions by catching one exception and throwing another.
- Create user-defined exceptions.
- Use the debugging feature `assert` to state conditions that should be true at a particular point in a method.
- Learn how `try`-with-resources can automatically release a resource when the `try` block terminates.

Outline

1. 11.1 Introduction
2. 11.2 Example: Divide by Zero without Exception Handling
3. 11.3 Example: Handling `ArithmeticException`s and `InputMismatchException`s
4. 11.4 When to Use Exception Handling
5. 11.5 Java Exception Hierarchy
6. 11.6 `finally` Block
7. 11.7 Stack Unwinding and Obtaining Information from an Exception
8. 11.8 Chained Exceptions
9. 11.9 Declaring New Exception Types
10. 11.10 Preconditions and Postconditions
11. 11.11 Assertions
12. 11.12 `try`-with-Resources: Automatic Resource Deallocation
13. 11.13 Wrap-Up

1. Summary
2. Self-Review Exercises
3. Answers to Self-Review Exercises
4. Exercises