Chapter 10

Exceptions

Exceptions

- Exception handling is an important aspect of object-oriented design
- Chapter 10 focuses on:
  - the purpose of exceptions
  - exception messages
  - the try-catch statement
  - propagating exceptions
  - the exception class hierarchy

© 2004 Pearson Addison-Wesley. All rights reserved

Exceptions

- An *exception* is an object that describes an unusual or erroneous situation
- Exceptions are *thrown* by a program, and may be *caught* and *handled* by another part of the program
- A program can be separated into a normal execution flow and an *exception execution flow*
- An *error* is also represented as an object in Java, but usually represents a unrecoverable situation and should not be caught

Exception Handling

- Java has a predefined set of exceptions and errors that can occur during execution
- A program can deal with an exception in one of three ways:
  - ignore it
  - handle it where it occurs
  - handle it an another place in the program
- The manner in which an exception is processed is an important design consideration

Exception Handling

- If an exception is ignored by the program, the program will terminate abnormally and produce an appropriate message
- The message includes a *call stack trace* that indicates the line on which the exception occurred
- The call stack trace also shows the method call trail that lead to the attempted execution of the offending line
  - The `getMessage` method returns a string explaining why the exception was thrown
  - The `printStackTrace` method prints the call stack track

© 2004 Pearson Addison-Wesley. All rights reserved

The try Statement

- To process an exception when it occurs, the line that throws the exception is executed within a *try block*
- A try block is followed by one or more *catch* clauses, which contain code to process an exception
- Each catch clause has an associated exception type and is called an *exception handler*
- When an exception occurs, processing continues at the first catch clause that matches the exception type

© 2004 Pearson Addison-Wesley. All rights reserved

The finally Clause

- A try statement can have an optional clause following the catch clauses, designated by the reserved word `finally`
- The statements in the finally clause always are executed
- If no exception is generated, the statements in the finally clause are executed after the statements in the try block complete
- If an exception is generated, the statements in the finally clause are executed after the statements in the appropriate catch clause complete

Exception Propagation

- An exception can be handled at a higher level if it is not appropriate to handle it where it occurs
- Exceptions *propagate* up through the method calling hierarchy until they are caught and handled or until they reach the level of the `main` method
- A try block that contains a call to a method in which an exception is thrown can be used to catch that exception

The throw Statement

- A programmer can define an exception by extending the `Exception` class or one of its descendants
- Exceptions are thrown using the *throw* statement
- Usually a throw statement is nested inside an if statement that evaluates the condition to see if the exception should be thrown

© 2004 Pearson Addison-Wesley. All rights reserved

Checked Exceptions

- An exception is either *checked* or *unchecked*
- A *checked exception* either must be caught by a method, or must be listed in the *throws clause* of any method that may throw or propagate it
- A throws clause is appended to the method header
- The compiler will issue an error if a checked exception is not handled appropriately

© 2004 Pearson Addison-Wesley. All rights reserved

Unchecked Exceptions

- An unchecked exception does not require explicit handling, though it could be processed that way
- The only unchecked exceptions in Java are objects of type `RuntimeException` or any of its descendants
- Errors are similar to `RuntimeException` and its descendants
  - Errors should not be caught
  - Errors to not require a throws clause

© 2004 Pearson Addison-Wesley. All rights reserved