chapter 0 Introduction to object-oriented software design
  0.1 What is a software system?
    0.1.1 Dealing with complexity: composition and abstraction
    0.1.2 Two aspects of a system: data and functionality
  0.2 Object-oriented systems
  0.3 Summary

chapter 1 Data abstraction: introductory concepts
  1.1 Objects
    1.1.1 Data and state
    1.1.2 Functionality: queries and commands
    1.1.3 Some cautions
  1.2 Values and types
    1.2.1 Values and types in Java
    1.2.2 Types and variables
  1.3 Classes
  1.4 An example
  1.5 Reference values: objects as properties of objects
  1.6 Overview of a complete system
    1.6.1 An example: testing a Counter
  1.7 Java in detail: identifiers and literals
    1.7.1 Identifiers
    1.7.2 Literals
  1.8 Java in detail: lexical structure
  1.9 Objects that “wrap” primitive values
  1.10 Java in detail: primitive types
  1.11 Java in detail: literals, the rest of the story
  1.12 Summary

chapter 2 Defining a simple class
  2.1 Object interaction: clients and servers
2.1.1 Specification and implementation

2.2 Defining a class: a simple counter
  2.2.1 Specifying the class features
  2.2.2 Static diagrams
  2.2.3 Invoking a method or constructor
  2.2.4 Interaction diagrams

2.3 Implementing the class Counter
  2.3.1 Implementing data
  2.3.2 Implementing functionality
  2.3.3 Documenting specification

2.4 Simple arithmetic expressions

2.5 The class TrafficSignal
  2.5.1 Named constants
  2.5.2 Implementing the class TrafficSignal

2.6 The class PlayingCard
  2.6.1 Constructor with parameters

2.7 Java in detail: arithmetic expressions

2.8 Java in detail: basic organizational structure

2.9 Java in detail: referring to classes in a different package, import statements, and the package java.lang

2.10 Java in detail: arithmetic expressions, the rest of the story

2.11 Summary

Chapter 3 Designing interacting classes

3.1 Designing with objects

3.2 A nim game example
  3.2.1 Implementing the class Pile
  3.2.2 Implementing the class Player

3.3 A maze game example
  3.3.1 Implementing the class Explorer

3.4 Local variables: another kind of method variable

3.5 Putting together a complete system
  3.5.1 Getting it started
  3.5.2 The method toString

3.6 Testing an implementation
  3.6.1 A class to test
  3.6.2 The TrafficSignalTest class
  3.6.3 The initiating class
  3.6.4 Writing the tests

3.7 Java in detail: static and final features
chapter 4 Conditions

4.1 Conditional statements
4.1.1 The if-then statement
4.1.2 The if-then-else statement
4.1.3 Compound statements

4.2 Boolean expressions

4.3 Handling multiple cases
4.3.1 Dangling else

4.4 Example: the class Date

4.5 Example: a combination lock

4.6 Java in detail: Boolean expressions
4.6.1 Reference equality and object equality
4.6.2 Boolean operators

4.7 Java in detail: logical operators and conditional expressions

4.8 Java in detail: the switch statement

4.9 Summary

chapter 5 Programming by contract

5.1 Programming by contract
5.1.1 Verifying preconditions: the assert statement

5.2 Further examples

5.3 Preconditions and postconditions: a summary of use

5.4 Enumeration classes

5.5 Summary

chapter 6 Testing

6.1 Functional testing and unit testing

6.2 Developing a test plan

6.3 The JUnit testing framework

6.4 An example

6.5 Do we need to test everything?

6.6 Summary
chapter 7  Building a text-based user interface

7.1  Relating the user interface and the model

7.2  Stream-based i/o

7.2.1  Writing standard output and reading standard input

7.3  An example: building an interface for Rectangle

7.3.1  Repeating actions: the while statement

7.3.2  Completing the createRectangle method

7.3.3  Finishing the user interface

7.3.4  Loop structure

7.4  A second example: using composition

7.4.1  Constructing a class by composition

7.4.2  Implementing the user interface

7.5  Java in detail: the do statement

7.6  Summary

chapter 8  The software life cycle: building a complete system

8.1  Software life cycle

8.2  Design of a system

8.2.1  Functional specification

8.2.2  Preliminary design

8.2.3  Relations between objects

8.2.4  Class specification

8.3  Implementing the system

8.3.1  The top-level

8.3.2  The class Pile

8.3.3  The class Player

8.3.4  The class Game

8.3.5  The class NimTUI

8.4  Summary

chapter 9  Specifying clients: interfaces

9.1  Modeling alternative implementations

9.2  Interfaces

9.2.1  Interface definition

9.2.2  Interface implementation

9.2.3  Subtyping

9.2.4  Putting it together

9.3  Clients and interfaces

9.4  Multiple inheritance and interface extension

9.5  Modifying the system: user vs. computer
<table>
<thead>
<tr>
<th>Chapter</th>
<th>Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.5</td>
<td>Player classes</td>
</tr>
<tr>
<td>9.5.2</td>
<td>The user interface</td>
</tr>
<tr>
<td>9.5.3</td>
<td>User interface – model interaction</td>
</tr>
<tr>
<td>9.6</td>
<td>Reclaiming lost ground: the strategy pattern</td>
</tr>
<tr>
<td>9.7</td>
<td>Java in detail: casting and <code>instanceof</code></td>
</tr>
<tr>
<td>9.8</td>
<td>Summary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>chapter 10</th>
<th>Class extension and inheritance</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Abstraction and classes</td>
</tr>
<tr>
<td>10.1.1</td>
<td>Class inheritance is single inheritance</td>
</tr>
<tr>
<td>10.2</td>
<td>Extension and inheritance</td>
</tr>
<tr>
<td>10.3</td>
<td>Constructors and subclasses</td>
</tr>
<tr>
<td>10.4</td>
<td>Overloading, overriding, and polymorphism</td>
</tr>
<tr>
<td>10.4.1</td>
<td>Method overloading</td>
</tr>
<tr>
<td>10.4.2</td>
<td>Method overriding</td>
</tr>
<tr>
<td>10.4.3</td>
<td>Overriding and contracts</td>
</tr>
<tr>
<td>10.5</td>
<td>Java in detail: feature accessibility</td>
</tr>
<tr>
<td>10.5.1</td>
<td>Protected features</td>
</tr>
<tr>
<td>10.5.2</td>
<td>Package private features</td>
</tr>
<tr>
<td>10.5.3</td>
<td>Inner classes</td>
</tr>
<tr>
<td>10.6</td>
<td>Java in detail: scoping rules</td>
</tr>
<tr>
<td>10.7</td>
<td>Summary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>chapter 11</th>
<th>Modeling with abstraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Abstract classes</td>
</tr>
<tr>
<td>11.1.1</td>
<td>Interfaces, abstract classes, and concrete classes</td>
</tr>
<tr>
<td>11.2</td>
<td>Specifying a class for extension</td>
</tr>
<tr>
<td>11.2.1</td>
<td>Planning for extension</td>
</tr>
<tr>
<td>11.3</td>
<td>Composition revisited</td>
</tr>
<tr>
<td>11.3.1</td>
<td>Extension and composition</td>
</tr>
<tr>
<td>11.3.2</td>
<td>Extension, composition, and reuse</td>
</tr>
<tr>
<td>11.3.3</td>
<td>Extension, composition, and modifying functionality</td>
</tr>
<tr>
<td>11.4</td>
<td>Extension and state</td>
</tr>
<tr>
<td>11.4.1</td>
<td>State as an object</td>
</tr>
<tr>
<td>11.5</td>
<td>Summary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>chapter 12</th>
<th>Lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>12.1</td>
<td>Containers</td>
</tr>
<tr>
<td>12.2</td>
<td>Lists</td>
</tr>
</tbody>
</table>
12.2.1 Structuring lists
12.2.2 Using generics to capture “list-ness”

12.3 List specification

12.4 Iteration
12.4.1 while loops and lists

12.5 Examples
12.5.1 Summing items on a list
12.5.2 Summing selected elements of a list
12.5.3 Finding the minimum
12.5.4 Determining if an object is on a List: searching the List
12.5.5 Removing duplicates

12.6 Loop structure: a summary

12.7 Java in detail: the for statement

12.8 What does “equal” mean?
12.8.1 Overriding equals
12.8.2 Problems when overriding equals

12.9 Implementing the interface List
12.9.1 The class AbstractList
12.9.2 The class DefaultList

12.10 Generic type checking is done at compile time

12.11 Summary

chapter 13 Arrays

13.1 Arrays
13.1.1 Creating arrays
13.1.2 Accessing array components

13.2 Examples

13.3 Constant arrays

13.4 Java in detail: static initializers

13.5 Arrays with array components
13.5.1 Creating multidimensional arrays
13.5.2 Accessing multidimensional array components
13.5.3 An example: tic-tac-toe

13.6 Types and arrays
13.6.1 Component types
13.6.2 Subtyping

13.7 An example: modeling lists with arrays

13.8 Java in detail: command line arguments

13.9 Java in detail: variable arity parameters
13.10 Summary

chapter 14 Sorting and searching
14.1 Ordering lists
14.2 Two simple sorts
14.2.1 Selection sort
14.2.2 Analysis of selection sort
14.2.3 Bubble sort
14.2.4 Generalizing the sort methods
14.2.5 The interfaces java.util.Comparator and java.lang.Comparable
14.3 Ordered lists
14.4 Binary search
14.4.1 Completing the search
14.4.2 Sequential search and binary search
14.5 Verifying correctness: using a loop invariant
14.6 Summary

chapter 15 Failures and exceptions
15.1 Failures
15.2 The Java exception mechanism
15.2.1 Exceptions as objects
15.2.2 Catching exceptions
15.2.3 Propagated exceptions
15.2.4 Checked and unchecked exceptions
15.3 Dealing with failure: using exceptions
15.3.1 Dealing with exceptions
15.3.2 Application defined exceptions
15.3.3 Dealing with logical errors
15.4 Exceptions and contracts
15.5 The CloneNotSupportedException and the interface Cloneable
15.6 Summary

chapter 16 Stream i/o
16.1 Data Streams
16.2 The java.io library
16.3 Input streams
16.3.1 Input byte streams
0.1 Input streams
0.1.1 Input byte streams
16.3.2 Input character streams
16.3.3 Input examples

16.4 Output streams
16.4.1 Output byte streams
16.4.2 Output character streams

16.5 The class File

16.6 Summary

Chapter 17 Building a graphical user interface

17.1 Event-driven interfaces

17.2 An introduction to Swing
  17.2.1 Components
  17.2.2 Containers

17.3 Creating a display
  17.3.1 The top-level frame
  17.3.2 The event dispatching thread
  17.3.3 Adding components: layout

17.4 Events: programming the user interface
  17.4.1 An example

17.5 A more complex example

17.6 Menus, dialogs, fonts, and graphics
  17.6.1 Menus and menu bars
  17.6.2 Basic dialogs
  17.6.3 Fonts
  17.6.4 Graphics

17.7 Some class features
  17.7.1 Component
  17.7.2 Container
  17.7.3 Window
  17.7.4 Frame
  17.7.5 JComponent
  17.7.6 JFrame

17.8 Summary

Chapter 18 Integrating user interface and model: the Model-View-Controller pattern

18.1 Model-View-Controller

18.2 Implementing MVC in Java
  18.2.1 The model
  18.2.2 The class Observable and interface Observer
18.2.3 An Observer
18.2.4 Observer, Observable, and subtyping
18.2.5 A simple view and controller
18.2.6 A graphic view
18.2.7 A logger as a view

18.3 Adding a graphical interface to the nim game
18.4 The MVC pattern and Swing components
18.5 Summary

chapter 19  Recursion

19.1 Recursion and iteration
  19.1.1 Iteration
  19.1.2 Recursion
  19.1.3 Some simple examples

19.2 Example: the tower puzzle
19.3 Quicksort
19.4 An inefficient algorithm
19.5 Indirect recursion
19.6 Backtracking
19.7 Object recursion
19.8 Summary

chapter 20  Generic structures

20.1 Multiple and bounded parameters
20.2 Wildcard types
  20.2.1 Generics and subtyping, revisited
  20.2.2 Wildcards
  20.2.3 Using wildcards
  20.2.4 Wildcards and generic methods

20.3 Erasure
  20.3.1 Restrictions due to erasure

20.4 Summary

chapter 21  Implementing lists: linked implementations

21.1 A linked List implementation
  21.1.1 Implementing LinkedList methods

21.2 Linked list variations
21.3 Doubly-linked lists
21.4 Limitations of linked structures
21.5 Dynamic storage allocation
21.6 Summary

Chapter 22 Iterators

22.1 Iterators
  22.1.1 Iterator classes
  22.1.2 Creating an iterator

22.2 List<Element> methods with iterators as arguments
  22.2.1 Improving LinkedList<Element>.LocalIterator
  22.2.2 Iterator extensions
  22.2.3 Iterators and list modification
  22.2.4 Internal iterators: forEachDo

22.3 Comparing implementations
22.4 The java.util Collection<Element> hierarchy
  22.4.1 Implementing for each

22.5 Summary

Supplement a Systems and software
  a.1 A model of a computer system
  a.2 Software tools

Supplement b Programming errors
  b.1 Errors in the programming process

Supplement c Applets
  c.1 Applets
    c.1.1 The class JApplet
    c.1.2 Applets as applications
    c.1.3 An example: a simple clock
    c.1.4 An example: an animated box

Supplement d Enumeration types: the rest of the story
  d.1 Enumeration types

Appendix i Compiling, executing, and documenting
  i.1 Compiling and running an application
    i.1.1 Compilation units
Niño-Hosch Contents xi

i.1.2 Package name and directory hierarchy
i.1.3 The environment variable CLASSPATH
i.1.4 Compiling and running

i.2 Generating documentation

appendix ii DrJava
appendix iii Controls and basic Latin: the first 128 Unicode characters

glossary
references

index