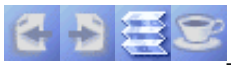 *Writing Java Programs*

Writing Java Programs: Table of Contents

If you want to learn about writing Java programs in a linear fashion, start at the first lesson listed in this Table Of Contents and walk your way through the lessons and pages of the "book". Each page has **Next** and **Previous** links at the bottom that guide you through the pages in the order listed here.

- Object–Oriented Programming Concepts: A Primer
  - What is an Object?
  - What are Messages?
  - What are Classes?
  - What is Inheritance?
  - Where Can I Get More Information?
- The Anatomy of a Java Application
  - Comparison of DateApp and "Hello World"
  - Defining a Class
  - The main() Method
  - An Introduction to Java Objects
    - Declaring, Instantiating and Initializing an Object
    - Static vs. Instance
  - Saving, Compiling and Running an Application
- The Nuts and Bolts of the Java Language
  - A Simple Class Definition
  - The main() Method
  - Introducing Exceptions
  - Variables and Data Types
  - Control Flow Statements
  - The Standard Input Stream
  - Operators
  - The Standard Output Stream
  - Character Data
  - Running the Application
- The String and StringBuffer Classes
  - Why Two String Classes?
  - Creating Strings and StringBuffers
  - Accessor Methods
    - More Accessor Methods
  - Modifying StringBuffers
  - Converting Objects to Strings
  - Converting Strings to Numbers
  - Strings and the Java Compiler
- Command Line Arguments
  - The Space Character Separates Command Line Arguments
  - Conventions
  - Parsing Command Line Arguments

- Threads of Control
  - What Are Threads?
  - A Simple Thread Example
  - Thread Attributes
    - Thread Body
      - The Clock Applet
    - Thread State
    - Thread Priority
    - Daemon Threads
    - Thread Group
  - Multithreaded Programs
    - Synchronization
    - Deadlock
    - Re–entrant Monitors
  - Summary
- Input and Output Streams
- Handling Errors using Exceptions
  - What's an Exception and Why Do I Care?
  - Your First Encounter with Java Exceptions
  - Java's Catch or Declare Requirement
  - Dealing with Exceptions
    - The Example
    - Catching and Handling Exceptions
      - The try Block
      - The catch Block(s)
      - The finally Block
      - Putting It All Together
    - Declaring the Exceptions Thrown by a Method
  - How to Throw Exceptions
    - The throw Statement
    - The Throwable Class and Its Subclasses
    - Creating Your Own Exception Classes
  - Runtime Exceptions––The Controversy

Writing Java Programs

Table of Contents

Writing Java Programs

The **Writing Java Programs** series covers everything you need to know about the Java language. In addition, some of the lessons describe the use of classes in the java.lang package.

Object–Oriented Programming Concepts: A Primer cuts through all the hype surrounding object–oriented technology and teaches you its core concepts: objects, messages, classes and inheritance. It's important to understand these key concepts before delving into the specifics of the Java language. This lesson also contains a bibliography for further reading on the subject. Feel free to skip this lesson if you are already familiar with object–oriented programming.

The Anatomy of a Java Application talks about the components of a stand–alone Java program in the context of a small application that prints the current date and time. Through this lesson, you will see classes and objects, two fundamental building blocks of object–oriented programming, in action within the context of a Java program.

The Nuts and Bolts of the Java Language introduces you to several components of the Java language by explaining a simple Java application line by line. You will learn about the syntax and semantics of the Java language, plus several features of the Java programming environment.

The String and StringBuffer Classes illustrates how to manipulate character data using the String and StringBuffer classes. Through this lesson you will also learn about accessor methods and how the compiler uses Strings and StringBuffers behind the scenes.

Command Line Arguments shows you how to accept and process arguments provided on the command line to a Java application. Note that the Java interpreter only passes command line arguments to Java *applications*––applets use parameters instead. For more information about applet parameters, see Communicating With the User .

Threads of Control discusses in detail the use of threads that enable your Java applications or applets to perform multiple tasks at the same time. This lesson describes when and why you might want to use threads, shows you how to create and manage threads and thread groups in your Java program, and shows you how to avoid common pitfalls such as deadlock and race conditions.

Handling Errors using Exceptions discusses how you can use Java's *exception* mechanism to handle errors in your programs. In this lesson you will learn what an exception is, how to throw and

catch exceptions, what to do with an exception once you've caught it, and how to best use the exception class hierarchy provided by the Java development environment.

Input and Output Streams [PENDING: this lesson is under construction.]

Table of Contents

Table of Contents

Writing Java Programs: End of Trail

You've reached the end of the "Writing Java Programs" trail.

Take a break –– have a steaming cup of java.

What next?

Once you've caught your breath, you have several choices of where to go next. You can go back to the Trail Map to see all of your choices, or you can go directly to one of the following popular trails:

Writing Applets: This is the starting point for learning everything about writing applets.

Creating a User Interface: Once you know how to create applications or applets, follow this trail to learn how to create their user interfaces.

The Java Development Environment: Follow this trail to learn what Java tools exist –– the packages (class libraries), compiler, interpreter, and so on –– and how to use them.

Table of Contents