

Seminar Abstract

Programming in Java™

© 1996–1999, 2005, 2009 Rex Jaeschke. All rights reserved.

Course Overview:

This course covers the Java language and its use in producing stand-alone programs called *applications*. This course or its equivalent is a prerequisite for the companion course *Programming with Java AWT*, which covers applets and windowing. Lab time is included. The course is not hardware or operating system-specific.

Course Length:

The length varies depending on the programming background of the participants, as follows:

# Days	Language Background
3	Fluent in C++
4	Fluent in some other OOP language
4	Fluent in C
5	Some other procedural language only

Goals:

Provided students meet the prerequisites, at the end of the course they should have a good understanding of the following:

- The goals of Java
- How Java is/is not like C and C++.
- The advantages of Java over C and C++.
- How to survive and thrive without an overt pointer type.
- The basic principles of OOP such as data hiding, encapsulation, inheritance, and polymorphism.

- Error handling via exceptions.
- What's available in the Java Class Library.

Who Should Attend:

Programmers and technical managers who are seriously interested in evaluating, and possibly using, Java for any purpose. Also, engineers and scientists currently using a procedural language, who want to reap many of the benefits of C/C++ without paying the significantly high price that programming in those languages extracts.

Prerequisites:

Knowledge of at least one procedural high-level language such as Fortran, Basic, or Pascal is assumed. Familiarity with C, C++, SmallTalk, or some other OOP language is a definite advantage but is not required. (See *Course Length* above.)

Those claiming fluency in C++ are expected to have a good working knowledge of the following topics: all the statements and operators, pointers and references, dynamic memory usage via new and delete, function inlining, function overloading, basic class design, operator overloading, single inheritance, virtual functions, and exception handling.

Those claiming fluency in some other OO-language are expected to have a good working knowledge of the following topics: basic class design, single inheritance, and exception handling.

Those claiming fluency in C are expected to have a good working knowledge of the following topics: all the statements and operators, pointers, dynamic memory usage via malloc and free, argument passing and return value handling, arrays, string handling, the new-style way of declaring and defining functions, and all aspects of structures.

Those claiming fluency in some procedural language are expected to have a good working knowledge of the following topics: variables, arrays, looping, operator precedence, type conversion, string processing, I/O, passing arguments to, and returning values from, a procedure, number system theory, bit manipulation, data representation, and group items (called records or structures in some languages).

Materials:

- *Programming in Java*— This manuscript was written specifically for teaching. It serves as a useful reference once the course has been completed.

Detailed Topics:

The main topics covered are:

- Basic Language Elements
- Looping and Testing
- Methods
- References, Strings, and Arrays

- Classes
- Inheritance
- Exception handling
- Input and Output
- Packages
- Interfaces