+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated 05/17/2024

Sign up

Training Advanced Multithreaded Programming C/C++20

3 days (21 hours)

Presentation

Our training in C++17 and 20 will help you improve the quality of your code and facilitate the management of your projects. You'll be able to easily implement advanced C++ features to guarantee the quality of your code.

In this course, aimed at developers, you'll learn how to use the main features of C++17 and 20, such as modules, concepts, generic lambdas, constexpr if statements and the extended standard library.

You will learn about the concepts, modeling processes and design patterns of multithreaded applications, as well as a wide range of tools to facilitate their design and development. An in-depth look will be taken at the primitives provided by the hardware and the system, and their impact in terms of performance.

In the same way, certain approaches from real-time programming, relevant to classic multithreaded applications and their optimization, will be studied.

Like all our training courses, this one will introduce you to the latest stable version and the new features of C++26.

As far as C++26 is concerned, we're waiting for the version to be stable before using it during this course, but the new features will be covered in our program.

Objectives

- Understand the fundamental concepts of C++ programming
- Master the latest features of C++17 and 20
- Use standard C++ libraries

- Mastering good C++ programming practices

Target audience

- Developers
- Architects

Prerequisites

- Knowledge of the Linux development environment
- Knowledge of C and C++ languages

Software requirements

An operational C++ development environment with a recent compiler.

Multithread C/C++ version 20 training program

Basic concepts of multithreaded programming

- Task and memory management model
- Principle of hardware operation

What's new in C++17

- Templates and Generic Code
- Lambda
- Attributes
- Syntax cleanup
- Cleaner multi-return and flow control
  - Library additions :
  - Data types
  - Invoke stuff
- File System TS v1
- New algorithms implemented
- Threading: Parallelism TS v1
- Features: swap, is_aggregate, has_unique_object_representations

Study of POSIX/Linux APIs

- Threads
- Synchronization primitives

- Shared memory & SYSV semaphore, the multiprocess approach
- Thread local storage

Algorithm parallelization

- Data partitioning
- Dependency and interaction modeling, pipeline definition
- Correspondence with architecture
- Data reduction
- Example of patterns

Hardware architecture

- Single/multiprocessor, single/multicore
- "Hyperthreading
- Cache management
- Memory management and NUMA

Scheduling

- Anatomy of a task
- Change of context and cost in performance
- Scheduling
- Linux scheduler
- Priority management
- Real-time queues
- Implementation of synchronization primitives.
- Reverse priority

Memory management

- Virtual memory: MMU, pages and TLB
- Multiprocess optimization
- Shared memory, TLS
- NUMA support

A look at latency and real-time

- Anatomy of a syscall
- Syscall latency
- Userland memory and I/O management

Study and practice of development tools

- Multithreaded debugging on gdb/lldb
- Kernel
  - perf
  - lockdep
  - stat
  - other
- The valgrind suite, intrinsic
- Analysis and control tools for memory and cache management.
- Scheduler analysis and control tools

Study and practice of development tools

- Thread safe containers
- Lockfree data structure
- Design patterns

Optimization

- Cache fighting & false sharing
- Restraint around locks
- Hyperthreading

Discover what's new in C++20

- The concepts
- Ranges
- The modules
- The coroutines
- Reflection
- The network
- 2D graphics (a la Cairo)

Existing C++ functionality removed

- Deleting trigraphs
- Deregistration
- Removal of the +++ operator.
- Removal of obsolete exception specifications
- Delete auto_ptr

What's new in future C++23

- Literal suffixes for std::size_t
- basic_string and std::basic_string_view
- The stacktrace library
- The <stdatomic.h> header

- Removing the list of unnecessary empty parameters () from lambda expressions

Companies concerned

This training course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives for the training to come, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Sanction

A certificate will be issued to each trainee who completes the course.