AMBIENT-IT

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

Updated 07/28/2023

Sign up

Modern C++ Training

3 days (21 hours)

Presentation

This modern C++ training course is designed for people who have already mastered the basics of the C++ language and want to improve their skills to become more robust, powerful and productive.

The C++ ecosystem has been evolving intensively since 2011, at the rate of a new standard every 3 years. On the hardware side, the multiplication of cores in recent years also means that we need to rethink the way we program. All this poses a real challenge in terms of adaptation, both in terms of assimilating new features and getting rid of obsolete practices.

This course has been designed to meet this challenge. By highlighting the fundamental principles that guide the evolution of the language, it becomes much easier to progress without getting lost in the complexity. And that means writing code that's both simpler and more reliable.

As with all our training courses, this one will introduce you to the latest version of C++, C++ 20.

Objectives

- Understand and implement modern C++ principles
- Familiarize yourself with C++ evolutions up to C++20
- Master advanced concepts of parallel and concurrent programming

Target audience

- Developers
- Architects

Prerequisites

- Basic knowledge of C++ (object and pointer concepts)

Modern C++ and Multithreading training program

Revision of language basics

- C++ language definition
- Functions: inline, static, const, constexpr, virtual, consteval, ...
- Plain Old Data (POD): struct vs class
- Declaration vs definition (One Definition Rule)
- STL: containers and algorithms
- Basic principles of templates, difference with constexpr
- Undefined behavior

Foundations of modern C++

- Introducing the C++ Core Guidelines
- Syntax changes (C++14, C++17, C++20)
- Robust resource management (RAII, smart pointers, scope guards)
- Notion of ownership, move semantics (std::move)
- Value/entity semantics
- Simplify complex initializations with lambdas
- More robust, expressive code with strong typedefs (aliases)

Advanced object-oriented programming

- Static and dynamic polymorphism
- Liskov substitution principle
- Inheritance vs. composition
- Abstract classes, PIMPL
- OOP without legacy
- SOLID principles

Notions of functional programming

- The lambdas
- Immutability (const) and pure functions
- Optional type with std::optional
- Composite type with std::variant
- Type erasure : std::string_view, std::span

Multithreading and scalability

- Parallelism vs. competition
- Race condition, impact of const keyword
- Protection of critical sections (std::mutex, std::condition_variable, ...)
- Scalability issues (Amdahl's law)
- Parallel STL algorithms (C++17)
- Lock free programming (std::atomic)
- Asynchronous programming: promise, future, coroutines
- Map, filter, reduce (Qt Concurrent)
- Thread Local Storage (thread_local)
- Detecting problems (tools available)
- Other approaches: OpenMP, CUDA, Boost.Interprocess, gRPC

Modern C++ developer tools

- Configuring your compiler
- Unit testing with Catch2
- Using CMake and CMake presets
- Automatic code formatting with clang-format
- Dependency management with Conan
- Code coverage with gcov / lcov
- Static analyzers
- Compiler explorer (Godbolt)

Companies concerned

This 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.

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.