C++ Training by Experts in High Performance Software

As a company we are focused on delivering top quality software specialising in high performance C++ development for the financial markets. Our software makes use of cutting edge C++ with innovative approaches to architecture to deliver resilient, low-latency, high-throughput systems. We are also active in the international standardisation of the C++ language with members in the British Standards Institute C++ panel.

An important aspect of standardisation in C++ is reaching out to the wider community to gain a better understanding of their pain points and also to ensure that changes to the standard are well understood and used correctly. This reinforcement and feedback helps make C++ a better language and ensures practitioners get the most out of the committee's work.

For this reason and also due to interest from clients, we are offering our advanced C++ training courses, developed by C++ committee members and delivered by a member of the C++ standards committee. This is a rare opportunity to not simply attend a C++ training course, but to actually learn and gain insights from someone actively involved in advancing the language.

So what makes our training courses unique?

- We actively use cutting edge C++ to solve hard technical problems in large complex systems — most companies who offer training simply have trainers with basic experience of the language and an understanding of the mechanics, but no real appreciation of the features they cover. Unlike most courses that only teach **How**, we also teach **Why** and more importantly **When** to use a feature or facility.
- We focus on teaching the right things in the right order to offer a holistic view on C++ — C++ is a huge language with many complexities and navigating those is hard. Too often developers see C++ as a giant toolbox of tricks to be used on a whim and fancy, and this often leads to overly complex code-bases with many differing styles as developers learn new tricks along the way. **We teach Idiomatic C++**, that is, C++ that makes use of **right techniques, and libraries, at the right times**. Idiomatic C++ leads to simpler code-bases with more consistent style and better reliability.
- We are actively involved in the standardisation of C++ — We know C++ extremely well and have decades of usage experience. We have been involved in how the language has evolved and also experienced that evolution first hand. As such we well aware of common pitfalls and the problems encountered through poor choice of techniques.

All our courses are delivered on-site by a C++ committee member ensuring you have the best opportunity to really learn C++ as it should be used. We offer **competitive rates** but our availability is limited so please be sure to engage us early to maximise your chances of getting the best times that suit you.

For more details, and to discuss availability and pricing, contact us at training@clearpool.io. Please state the course or courses you are interested in and approximate numbers of attendees. Our pricing aims to make these courses accessible and support our aims of promoting high quality idiomatic C++ use in the industry, while also gaining feedback on the level and usage patterns of existing practitioners.

Guide to Training Courses

We currently offer five courses of varying durations but typically lasting 3 to 5 days. Some courses offer an accelerated program where content is delivered at a faster pace. Attendees of accelerated courses are expected to do additional work themselves outside of training times to obtain the full value from the course.

Courses are delivered on-site using our pre-configured development set-ups. This ensures consistency of platform and learning experience. There is a only minimal set-up required to make a host machine ready and we will provide you with all the details several days beforehand. In most cases attendees can carry out the necessary updates themselves in a matter of minutes as we make use of technologies such as Docker. Some time is allowed at the start of the course to review everyone's set-up. An advantage of this approach is that all attendees can continue to use this pre-configured environment after completion of the course for further experimentation and learning.

Courses are delivered as a mixture of slides and exercises with attendees given both a copy of the slides, and in some cases, a reference book that can be used to find additional details. One of the learning outcomes is to encourage intelligent use of available resources and the course material is extensively referenced to facilitate wider reading. The only hosting requirements for the courses are therefore a pre-configured machine for each attendee and a suitable slide presentation medium.

The goal of these courses is to teach **idiomatic C++** and as such the focus is on sensible use of features, not simply summarising how features or libraries work. C++ provides many tools and more often than not they are used inappropriately in overly-complex ways resulting in large, difficult to maintain code-bases. Each topic covered in the course not only introduces and describes the relevant feature or library but also offers advice and when and why they should be used. In fact the course material is driven by idiomatic usage of C++ over features available in C++.

The five courses currently being offered are summarised below. The following pages offer more details.

Course	Description
Course 1: Modern, Idiomatic C++	This course focuses on teaching modern, idiomatic C++11/C++14. Any competent developer should be able to complete the course. This course will suit most developers.
Course 2: Advanced Idiomatic C++	This advanced course moves beyond the topics covered in Modern, Idiomatic C++ to focus more on library writing and the delivery of high quality reusable components. This course also looks at techniques to exploit concurrency and higher level architectural concerns in large C++ systems.
Course 3: Idiomatic C++11/C++14	Designed mainly as a language facility overview for those developers that already feel they are well-versed in idiomatic C++98/C++03. This course aims to survey the recent changes in the language and offer constructive advice on their use. For most developers Modern, Idiomatic C++ will offer a more holistic introduction.
Course 4: High Performance C++	This course is for developers working on very high performance or large scale complex systems. C++ developers in Financial markets, for example, would greatly benefit. It covers topics such as algorithmic architecture, exploiting concurrency, low latency and high throughput. This course requires attendees to already be comfortable with idiomatic C++11/C++14 and preferably experience of complex systems.
Course 5: Modern C++ with Qt5	Qt is the leading cross-platform GUI framework for C++. Qt itself has evolved over time and uses an old style object hierarchy. This course aims to equip developers with sufficient knowledge to leverage the power of Qt while making as much use of modern C++11/C++14 as possible.

Course 1: Modern, Idiomatic C++

Target Audience Developers who are competent in another language and who possibly have some previous exposure to C++, but this is not required. The key thing is they are comfortable with programming concepts in general.

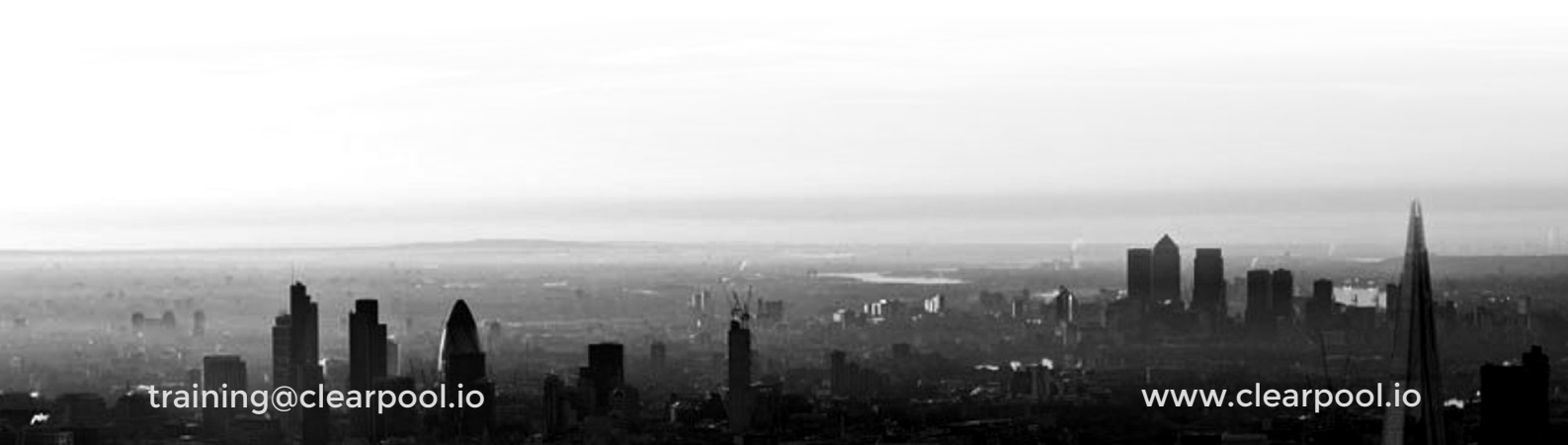
Delivery On-site teaching with Slides, Reference Book and Hands-on exercises

Duration Standard course: 5 days
Shortened course: 3 days (some topics are removed)
Accelerated course: 3 days (all topics covered but in less depth)

Expected Outcomes On completion of the course you will be equipped with sufficient knowledge to write safe and usable C++ code. You will understand how to manage object lifetimes and provide consistent and safe behaviour in the presence of exceptions. You will be able to select the right tool for the right job, make judicious use of appropriate coupling mechanisms and know how to avoid common mistakes. Lastly you will know where to look next to find additional libraries and techniques.

Sample Topics Covered

- Iterators and Containers
- Algorithms
- Strings and Regular Expressions
- Tuples
- Exception safety and guarantees
- Const-correctness
- Lifetime management (RAII)
- Smart Pointers
- Immutable objects
- Multi-paradigm design: procedural, object oriented, generic
- Static polymorphism and templates
- Writing C++ for loose coupling
- The factory mentality
- Functional facilities: std::function and lambdas
- Standard library overview
  - More Containers
  - Random Numbers
  - Chrono
- Boost library introduction
- Networking TS / Asio Introduction
- Overview of C++ Standardisation process



Course 2: Advanced Idiomatic C++

Target Audience Developers who have completed Modern, Idiomatic C++ or who are comfortable with the content. Have prior experience with idiomatic usage of C++11/C++14. If unsure please contact us to discuss.

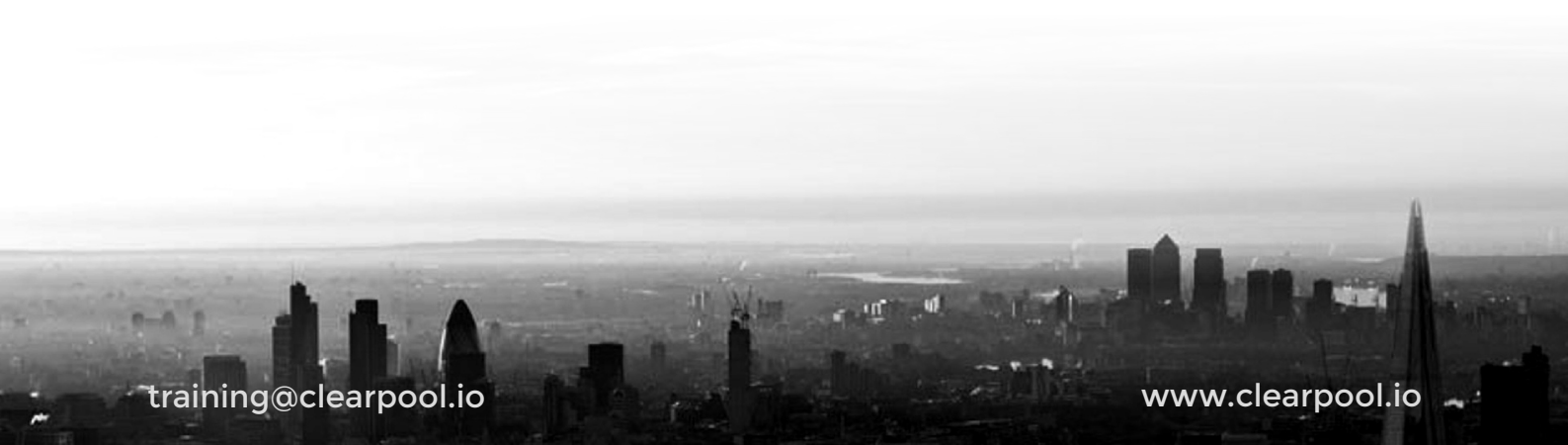
Delivery On-site teaching with Slides, Reference Book and Hands-on exercises

Duration Standard course: 5 days
Shortened course: 3 days (some topics are removed)

Expected Outcomes On completion of the course you will be equipped with sufficient knowledge to understand key techniques in designing and building re-usable libraries in C++ for use within your company. You will appreciate the importance of well-designed interfaces and be able to discern poorly designed libraries from good ones. You will also have gained an appreciation of concurrency facilities in C++ and know where to look next to learn more.

Sample Topics Covered

- Tools for library writers
  - SFINAE
  - Type traits
  - Inline namespaces
  - Memory Management
  - Operator overloading
- Value and Reference Semantics
- Move semantics
- Generic programming
- Lifetime management
- Error handling, Exceptions and Contracts
- Library writing guidelines
- Introduction to Input/Output extension
- C++ memory model
- Introduction to concurrency facilities
- Introduction to Networking TS / Asio
- Future direction of C++ Standardisation



Course 3: Idiomatic C++11/C++14

Target Audience Competent developers who are already familiar with **idiomatic** C++98 or C++ 03. The primary goal of this course is to fast track the adoption of best practices when adopting C++11/C++14 for people who are already experienced with best practices in C++98/C++03.

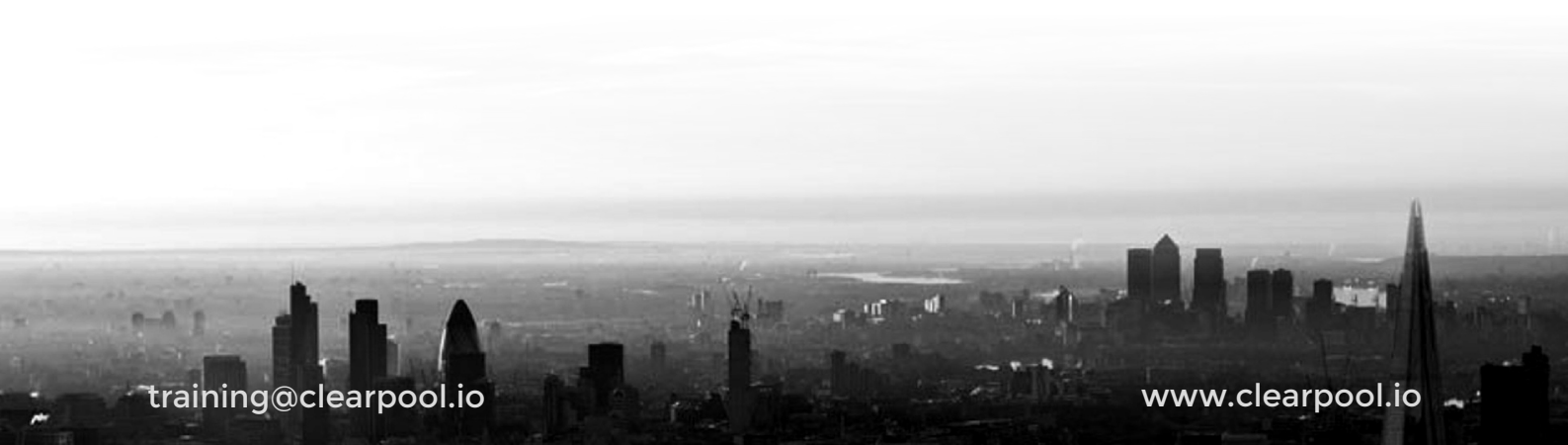
Delivery On-site teaching with Slides, Reference Book and Hands-on exercises

Duration Standard course: 3 days

Expected Outcomes Completion of the course will bring an appreciation of how best to utilise the new features in C++11 and C++14. Some things are simpler, and some things require a change in style or thinking to make best use of them in the language. Attendees should come away from the course with a solid overview of new features and idioms, along with the confidence to know when to use them.

Sample Topics Covered

- auto
- New function syntax
- New style for
- constexpr
- Lambdas
- Move semantics
- Rvalue and xvalue references
- Initialisation
- decltype
- noexcept
- Tuples
- Smart pointers
- Template aliases
- Inline namespaces
- Strongly type enums
- Variadic templates and forwarding
- The C++ memory model
- Overview of new libraries in C++11/C++14



Course 4: High Performance C++

Target Audience Will ideally have completed one of Modern, Idiomatic C++, Advanced Idiomatic C++ or Idiomatic C++11/C++14. If not then it is expected that you have prior experience with idiomatic usage of C++11/C++14. If unsure please contact us to discuss.
This course will be particularly attractive to developers working on very low-latency or very high throughput systems, for example, developers in the Financial markets will benefit greatly from this course. Covering not just C++ but also architecture and design this course distils many years of experience building high performance systems.

Delivery On-site teaching with Slides and Hands-on exercises

Duration Standard course: 3 days
Extended course: 5 days

Expected Outcomes On completion of the course you will be equipped with sufficient knowledge to understand key techniques in designing and building re-usable libraries in C++ for use within your company. You will appreciate the importance of well-designed interfaces and be able to discern poorly designed libraries from good ones. You will also have gained an appreciation of concurrency facilities in C++ and know your where to look next to learn more.

Sample Topics Covered

- Static Polymorphism using Templates
- Generic Programming
- Algorithmic Architecture
- Low Latency and High Throughput
- Concurrency in C++
- Message Passing
- Lambdas
- Introduction to Networking TS / Asio Executors
- Atomics
- Introduction to Boost.Lockfree
- Measuring performance
- Impact of Hardware

Course 5: Modern C++ with Qt5

Target Audience Competent C++ developers (preferably with C+11 experience) who wish to gain an introduction to user interface development using Qt5 as their base framework

Delivery On-site teaching with Slides and Hands-on exercises

Duration Standard course: 5 days

Expected Outcomes The primary goal of this course is to show how modern C++ can be used with a monolithic object-oriented user interface framework like Qt and minimise the use of old idioms and adopt good C++ practices were possible. On completion attendees will be able to write reasonably complex user interface applications.

Sample Topics Covered

- Principles of User Interface development
- Introduction to the Qt5 toolchain and MOC
- Introduction to using cuppa with Qt5
- Qt5 Overview
- Introduction to Lambdas
- Mixing Modern C++ with Qt
- Lifetime management
- Initialisation
- Overview of interface layouts
- Overview of basic widgets
- Overview of model/view framework
- QTreeView
- Integrating the Qt event loop with threads and Boost.Asio
- Overview of approaches to styling
- Introduction to Boost.Locale