Stand 11.05.2026

Embedded C++ für Fortgeschrittene: Objektorientierte Programmierung für Mikrocontroller mit C++/EC++ - Live-Online-Training

Die steigende Komplexität in Embedded-Softwareapplikationen und die immer leistungsfähigere Hardware erlauben inzwischen, C++ in steigendem Maße auch in Embedded-Systemen einzusetzen. Dabei sind abhängig von der Applikation Qualitätsmerkmale wie Sicherheit (Safety und Security), Performance, Ressourcenverbrauch und andere zu berücksichtigen.

**Ziele -
Ihr Nutzen**

Sie kennen die Anwendung und die Effizienz fortgeschrittener C++ Konstrukte (Namespaces, Templates, Exception Handling, Runtime Type Identification, New Style Casts, Mehrfachvererbung, Speichermanagement).

Sie sind in der Lage, über die Verwendung dieser Konstrukte in Ihrer Applikation fundiert zu entscheiden.

Sie haben einen Überblick über die Elemente und Mechanismen der STL (Standard Template Library) und können diese bereits einsetzen.

Sie können Patterns (State-Pattern, Singleton-Pattern, Observer-Pattern, Smart-Pointer-Pattern und Layer-Pattern) auf Ihre Applikationen hin adaptieren und dort implementieren.

Teilnehmer

Der EC++ Kurs für Fortgeschrittene richtet sich an Programmierer, Software-Entwickler, Software-Designer und Software-Architekten, die fortgeschrittene C++ Konstrukte in der Embedded-Softwareentwicklung einsetzen.

Voraussetzungen

Objektorientierte C++ Grundlagen sind zwingend erforderlich, UML-Grundlagen von Vorteil.

Live Online Training

07.09. – 10.09.2026	2.600,00 €	4 Tage

* Preis je Teilnehmer, in Euro zzgl. USt.

Anmeldecode: L-EC++FOR

Präsenz-Training - Deutsch

Termin	Dauer
08.06. – 11.06.2026	4 Tage

Live-Online - Englisch

Dauer
4 Tage

Präsenz-Training - Englisch

MICRO CONSULT

Stand 11.05.2026

Termin	Dauer
08.06. – 11.06.2026	4 Tage

Embedded C++ für Fortgeschrittene: Objektorientierte Programmierung für Mikrocontroller mit C++/EC++ - Live-Online-Training

Inhalt

C++ für Embedded-Applikationen

- Historie
- Spezielle Qualitätsanforderungen an Embedded-Software
- Empfehlungen und Regelwerke (Codier-Richtlinien)
- C++ Compiler-Prinzipien
- Ausblicke: C++ Idiome und Clean Code Development
- Praxistipp mit wichtigen Referenzen

Zusammenfassung grundlegender C++ Konstrukte mit Effizienzbetrachtungen

- Klasse und Objekt
- Bestandteile von Klassen
- Modifizierer für Daten, Funktionen und Objekte
- Klassen-Funktionszeiger am Beispiel Zustandsfolgeautomaten-Implementierung
- Klassenrelationen (Assoziation, Aggregation, Komposition und Vererbung)
- Virtuelle Funktionen und Interfaces (Realisierung und Zugriff)
- Erweiterte C++ Konstrukte

Namespaces mit Effizienzbetrachtungen

- Verwendung von (verschachtelten) Namespaces im Programmcode
- Namespace Alias, anonymer Namespace, Koenig-Lookup, inline Namespace
- Abbildung der Software-Architektur im Programmcode
- C++ Standard-Namespace std
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Basierend auf der Architektur arbeiten Sie Namespaces in den bestehenden Programmcode ein

Einfachvererbung und Mehrfachvererbung mit Effizienzbetrachtungen

- Programmierung der Einfach- und Mehrfachvererbung (mit Interfaces)
- Problemsituationen und Lösungen bei der Mehrfachvererbung
- Virtuelle Vererbung
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie verwenden und programmieren die Mehrfachvererbung, wahlweise virtuell

Exception Handling mit Effizienzbetrachtungen

- Exception Handling - Erläuterung und Programmierung
- Exception-Klassen und -Hierarchien
- Benutzer-Exceptions
- C++ System-Exceptions
- Verschachteltes Exception Handling
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie binden Exception Handling in die bestehende Applikation ein

Speichermanagement mit Effizienzbetrachtungen

- Speichersegmente (BSS, Stack, Heap) für Objekte im Vergleich
- Dynamisches Speichermanagement mit new und delete mit und ohne Exception Handling
- Operatorüberladung von new und delete
- Pool-Allocation Pattern
- Placement new
- Risiken und Stolpersteine vermeiden
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie erzeugen und löschen Objekte dynamisch auf dem Heap

Runtime Type Identification (RTTI) mit Effizienzbetrachtungen

- Erläuterung und Programmierung von RTTI

- type_info Klasse
- Konsequenzen beim Einsatz
- Bezug zu Exception Handling und New Style Casts
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie verwenden RTTI zur Klassenidentifikation zur Laufzeit

Typkonvertierung mit New Style Casts mit Effizienzbetrachtungen

- Static, dynamic, const und reinterpret Cast
- Die richtige Wahl beim Einsatz
- Bezug zu RTTI und Exception Handling
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software

Templates mit Effizienzbetrachtungen

- Template-Funktionen
- Template-Klasse und -Objekt
- Template-Parameter und -Alias
- Vererbung und Interfaces mit Template-Klassen
- Praxistipps: Statische versus dynamische Polymorphie
- CRTP (Curiously Recurring Template Pattern)
- Template-Spezialisierung und (implizite versus explizite) -Instanziierung
- Type Traits und Concepts
- Variadische Template-Funktionen und -Klassen
- Perfekt Forward
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie programmieren eine Template-Klasse zur Anwendung im Observer-Pattern-Kontext der Applikation

Smart Pointer, sinnvoller Einsatz mit Effizienzbetrachtungen

- Besonderheiten und Varianten von Smart Pointer
- Programmierung eigener Smart Pointer
- C++ Smart Pointer
- Lambda-Funktionen
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software

C++ Standard-Bibliothek: Container, Iteratoren und Algorithmen, sinnvoller Einsatz und Effizienzbetrachtungen

- Grundlegende Konzeptübersicht
- Container, Iteratoren, Adapter, Algorithmen
- Differenzierung und Vorstellung unterschiedlicher Container
- Container für sequenzielle, sortierende und spezielle Anwendungsfälle
- Funktionsobjekte (Funktoren)
- Lambda-Funktionen
- Allocator-Klasse
- Assembler-, Speicher- und Laufzeit-Analysen und -Optimierungen
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie verwenden eine Container-Klasse im Observer-Pattern-Kontext der Applikation

Callback

- Embedded-Software-Architekturrichtlinien
- Embedded-Software-Qualitätsmerkmale
- Software-Layer-Pattern: Embedded-Software-Schichtenarchitektur
- Synchrone und asynchrone Software-Architektur
- Unidirektionale und bidirektionale Kommunikation
- Callback-Struktur mit und ohne Betriebsmittel eines Betriebssystems

Hardware-Treiber und Interrupts mit C++

- Objektorientierte Konzepte und Programmierung von Standard-Peripherietreibern
- Objektorientierte Konzepte und Programmierung von Interrupt-Behandlungen
- Registerbank-Zugriff
- Callback-Strukturen im Interrupt-Kontext
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software
- Übung: Sie binden einen Hardwaretreiber und Interrupt-Service in die Applikation ein

Ausgewählte C++ Bibliothekselemente

- std::string und std::string_view
- iosteam und iomanip
- std::stringstream
- std::function, std::optional, std::variant und std::any
- Anwendungsbeispiele und Empfehlungen zur Verwendung in Embedded-Software

Praktische Übungen im Workshop

- Für die durchgängige Übung (Uhrenapplikation) verwenden Sie die STM32CubeIDE Entwicklungsumgebung zusammen mit den STM32 NUCLEO-F746ZG Evaluation Board, basierend auf einem Arm Cortex®-M7 Mikrocontroller.

MicroConsult PLUS:

- Ihre Übungen und die von MicroConsult entwickelten Lösungen aus dem Workshop stellen wir für Sie zum Download bereit.
- Sie erhalten alle C++ Beispiele in elektronischer Form und können diese sehr einfach für Ihre Entwicklungsumgebung anpassen.
- Sie erhalten eine Zusammenfassung von skalierbaren Empfehlungen zur Anwendung von C++ in Embedded-Software als Checkliste.
- Sie erhalten zudem eine Tool- und Software-Komponentenübersicht für die Entwicklung von Embedded-Software.
- Sie bekommen hilfreiche Notationsübersichten für UML und SysML.