MICRO CONSULT

Stand 16.06.2026

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

Meistern Sie die Herausforderung bei der Softwareentwicklung für Hard-Realtime-Systeme: Die heutigen Embedded-Systeme mit komplexen Mikrocontroller- und Prozessorarchitekturen enthalten immer mehr Software, die aber in immer kürzerer Zeit geplant und realisiert werden muss. Häufig sind Vorgaben zu erfüllen, in denen Normen und sicherheitskritische Aspekte berücksichtigt werden müssen. Echtzeitfähigkeit, Wiederverwendbarkeit, Anpassbarkeit an veränderte Rahmenbedingungen und leichte Lesbarkeit der Software spielen eine immer größere Rolle.

**Ziele -
Ihr Nutzen**

Sie kennen nach dem EC++ Workshop die Möglichkeiten, die ein objektorientierter Ansatz bietet, um auch in der Embedded-Welt in kürzerer Zeit qualitativ hochwertige Software zu entwickeln.

Der vermittelte Überblick über den gesamten Entwicklungsprozess von Embedded-Realtime-Systemen - Software-Analyse, Software-Design, Software-Implementierung und Unit-Test - gibt Ihnen das Rüstzeug, um Projekte mit EC++ zu erstellen.

Sie wissen, wie Sie das Design von Software-Systemen mithilfe der UML und der Implementierung in der Programmiersprache C++ gestalten.

Damit verfügen Sie über die Wissensbasis zur Erfüllung besonderer Anforderungen an die Software-Qualität von Embedded-Systemen, z.B. im Hinblick auf Laufzeit und Codeeffizienz.

Ferner vermeiden Sie mithilfe von Programmierrichtlinien, z.B. dem MISRA-C++ Standard, frühzeitig Programmierfehler.

Sie entwickeln mit modernen objektorientierten Techniken qualitativ hochwertige und komplexe Softwaresysteme und sparen dabei Zeit und Geld.

Teilnehmer

Der EC++ Kurs richtet sich an Programmierer, Software-Entwickler, Software-Designer und Software-Architekten, die C++ für Embedded-Softwareapplikationen unter Verwendung objektorientierter Konzepte einsetzen.

Voraussetzungen

Sie sollten über solide C-Programmierkenntnisse verfügen; Mikrocontroller-Grundkenntnisse sind 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++

Präsenz-Training - Deutsch

Termin	Dauer

07.12. – 10.12.2026 4 Tage

Live-Online - Englisch

Dauer
4 Tage

Präsenz-Training - Englisch

Termin	Dauer
22.06. – 25.06.2026	4 Tage

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

Inhalt

Einführung in die objektorientierte Begriffswelt

- Klasse und Objekt
- Attribut, Operation, Methode
- Kapselung
- Relation (Beziehung) zwischen Klassen
- Nutzen und Vorteile der objektorientierten Softwareentwicklung
- Demonstration des objektorientierten Ansatzes an einem Hardwaretreiber-Beispiel

Objektorientiertes Programmieren in C

- Klassen, Attribute, Operationen und Objekte
- Möglichkeiten der Kapselung in C
- Assoziation, Aggregation, Komposition und Vererbung
- Praxistipps mit bewährten Umsetzungsmöglichkeiten

Einführung in die Programmiersprache C++ (EC++)

- Migration von objektorientierter Programmierung mit C zu C++/EC++
- Klasse, Attribute, Operationen und Objekte
- Möglichkeiten der Kapselung in C++
- Konstruktoren, Destruktor, Überladen
- Modifizierer für Daten, Operationen und Objekte
- Speichermanagement für Objekte
- Objekt-Initialisierung
- Namespaces
- Templates
- C++ New Style Casts
- Weitere C++ spezifische Mechanismen und Konstrukte
- Assembleranalysen von C++ Konstrukten
- Speicher- und laufzeitorientierte Einschätzung und Optimierung
- Vergleich zu C
- Verbesserung von Qualitätsmerkmalen wie Wartbarkeit und Wiederverwendbarkeit
- Risiken erkennen, Stolpersteine vermeiden
- Übung: Sie implementieren, instanziieren und testen eine Klasse

Relationen zwischen Klassen und deren Programmierung mit C++

- Assoziation
- Aggregation
- Komposition
- Vererbung
- Dependency
- Richtige Relationsauswahl im Design treffen
- Varianten der Implementierung mit C++
- Assembleranalysen von C++ Konstrukten
- Speicher- und laufzeitorientierte Einschätzung und Optimierung
- Verbesserung von Qualitätsmerkmalen wie Wartbarkeit und Wiederverwendbarkeit
- Übung: Sie implementieren die Relationen Assoziation, Vererbung und Komposition in der Applikation und bei

der Anwendung des Builder-Patterns

Virtuelle Methoden und Interfaces mit C++

- Überschreiben von Funktionen / Operationen
- Dynamische (späte) Bindung
- Rein virtuelle Funktionen/ Operationen
- Polymorphismus
- Von der abstrakten Klasse zur Interface-Klasse
- Software-Architektur mit Interfaces
- Nutzen, Vorteile und Nachteile von Interface-Klassen
- Implementierung mit C++
- Assembleranalysen von C++ Konstrukten
- Speicher- und laufzeitorientierte Einschätzung und Optimierung
- Verbesserung von Qualitätsmerkmalen wie Wartbarkeit und Wiederverwendbarkeit
- Übung: Sie programmieren eine Interface-Klasse mit rein virtuellen Funktionen, implementieren diese und greifen darauf zu

C++ Klassenbibliotheken

- Statische versus dynamische Klassenbibliotheken
- Erzeugung und Einbindung von Klassenbibliotheken
- Vor- und Nachteile bei der Verwendung in Embedded-Software
- Demonstration eines Beispiels

Zustandsautomaten mit C++

- Prinzipielle Möglichkeiten der Implementierung
- Vorstellung der Tabellen-basierten Abarbeitung von Zustandsautomaten
- State-Pattern und objektorientiertes State-Pattern
- Demonstration eines Beispiels

Callback-Strukturen

- Typische Embedded-SW-Architekturen und deren Darstellung (Schichtenmodell, Interfaces)
- Architekturrichtlinien
- Kommunikationsmöglichkeiten zwischen Architekturelementen
- Objektorientiere Callback-Strukturen mit Callback-Objekten
- Demonstration eines Beispiels
- Übung: Sie programmieren eine Callback-Struktur mit Callback-Objekt-Registrierung in der Übungsapplikation

Hardwaretreiber und Interrupts mit C++

- Treiber- und Interrupt-Richtlinien
- Hardwaretreiber mit Peripherie-Registerzugriff objektorientiert in C++
- Interrupt-Konzepte und Interrupt-Service Routinen objektorientiert mit C++
- Verbesserung von Qualitätsmerkmalen wie Wartbarkeit und Wiederverwendbarkeit
- Übung: Sie wenden einen objektorientierten Timer-Treiber zusammen mit einem Timer-Interrupt in Ihrer Übungsapplikation an

Übersicht Embedded-Software-Test

- Software-Implementierung und -Test
- Testprozess
- Checkliste zum Test objektorientierter Software
- Demonstration eines Beispiels mit GoogleTest™

Grundlegende Notationen mit der UML (Unified Modeling Language)

- Darstellung von Klassen, Objekten und Relationen mit der UML
- Use-Case-, Sequenz-, Aktivitäten- und Zustandsfolge-Diagramm
- Demonstration der Übung im UML-Modell - Aspekt: Diagramme

Von der Projektplanung bis zur Implementierung

- Software-Entwicklungsprozess
- Qualität von Embedded-Softwaresystemen
- Praxisgerechte Modellierung einer Embedded-Applikation mit der UML
- Demonstration der Übung im UML-Modell - Aspekt: Modellentstehung und Aufbau

Praktische Übungen im Workshop

- Für die durchgängige Übung (Uhrenapplikation) verwenden Sie das Arm Keil MDK (Microcontroller Development Kit) zusammen mit einer realen Hardware basierend auf einem Arm Cortex® M3 oder M7 Mikrocontroller bzw.

einem entsprechenden Simulator. Die Übung besteht sowohl aus dem Programmcode als auch dem zugehörigen UML-Modell.

Ausblicke

- Historie und Weiterentwicklung von C++
- Übersicht zu fortgeschrittenen C++ Mechanismen
- Embedded- und generische C++ Codierrichtlinien
- Interessante Internet-Links
- C++ Idiome
- Clean Code Development

Praktische Übungen im Workshop

- Für die durchgängige Übung (Uhrenapplikation) verwenden Sie das Arm Keil MDK (Microcontroller Development Kit) zusammen mit einer realen Hardware basierend auf einem Arm Cortex® M3 oder M7 Mikrocontroller bzw. einem entsprechenden Simulator. Die Übung besteht sowohl aus dem Programmcode als auch dem zugehörigen UML-Modell.

MicroConsult PLUS:

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