FEABHAS

C++ for Embedded Developers

Course category	C++ Training Courses
Training area	Programming Languages
Course code	C++-501
Duration	5 days
Course date	25th March 2019, 3rd June 2019, 16th September 2019, 9th December 2019
Price exc VAT	£2650.00

This course introduces the C++ language for use on real-time and embedded applications. The course highlights areas of concern for real-time and embedded development. The focus is on developing core object-oriented programming skills and understanding of how to build effective, maintainable and efficient C++ programs.

Attendees perform hands-on embedded programming, on target hardware, during course practicals. Approximately 50% of the course is given over to practical work.

Course objectives:

- To provide a solid understanding of the essentials of the C++ programming language.
- To give you practical experience of writing C++ for real-time and embedded systems.
- To give you the confidence to apply these new concepts to your next real-time project.

Delegates will learn:

- The core C++ syntax and semantics
- How to access hardware in the language
- How to program interrupt handlers in C++
- About memory and performance issues associated with C++
- How real time operating systems (RTOS) affect the use of the language

Pre-requisites:

- A good working knowledge of C

Who should attend:

This course is designed for real-time engineers who are embarking on a project using C++ for the first time. It is also targeted at developers currently reluctant to move from C to C++ as they believe it poses too great an overhead. This course will clearly demonstrate both the strengths and weaknesses of C++ versus C.

Duration:

- Five days.

Course materials:

- Delegate handbook
- Delegate workbook
- Delegate datakey

Course workshop:

This course makes use of target hardware during the real-time practical exercises. The board targeted is an ARM Cortex-M based MCU which gives attendees a real sense of embedded application development.

Hello World!:

- How C++ relates to C
- I/O mechanisms

Declarations and definitions:

- Object lifetime and scope

Principles of Object-Oriented Design:

- Modularisation
- Objects and messaging
- Dealing with complexity

Creating objects

- Classes and instances
- Member variables and member functions

Initialising objects

- Constructors and destructors
- Objects as function parameters
- In, Out and InOut parameters
- References
- const correctness

Structuring code

- Header files

Hardware manipulation

- Memory-mapped hardware access
- Object-oriented hardware abstraction

Connecting objects

- 1:1, 1:N associations

Building complex objects

- Composition of objects

Resource management

- RAII / RDID
- Copy constructors and assignment operators
- The Rule of the Big Three

Specialisation

- Why do we have inheritance in OO design?

Creating substitutable objects

- Inheritance
- Constructing derived objects
- Accessing base class members

Overriding

- Overriding and hiding
- Dynamic polymorphism
- Virtual functions and their implementation

Abstract base classes

- The Single Responsibility Principle
- Extension of interface
- Safe down-casting

Realising interfaces

- The pure virtual class
- Interface segregation
- Cross-casting

Templates

- Template functions
- Template classes

Exception handling

- The exception handling mechanism
- Exception objects

Principles of concurrency

- Scheduling patterns

Creating concurrent objects

- Thread-Is-Polymorphic-Object pattern

Mutual exclusion

- Race conditions and their impact
- Mutex classes
- The scope-locked idiom

Thread synchronisation

- Signals
- Condition objects
- The Monitor pattern

Feabhas Ltd - UK Office 15-17 Lotmead Business Park, Wanborough, Swindon, SN4 0UY, UK •

+44 (0) 1793 792909 • info@feabhas.com • www.feabhas.com