Course C++-501: C++ for Embedded Developers

Course Description:

This course introduces the C++ language for use on real-time and embedded applications. The first part of the course focuses on the language itself, highlighting areas of concern for real-time and embedded development. The latter part of the course covers the application of C++ to real-time systems including interrupt handling and concurrency issues. If a C++ course fails to cover these fundamental issues you may be left feeling you still have a lot to learn.

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

Overview:

A five day course that provides a general overview of C++ on the first three days and then tackles real-time issues on the remaining two days. 50% of the course is spent on 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 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

Related Courses:

- OO-503 Real-Time Systems Design with UML 2.0
- OO-303 Applying Real-Time UML
- AC++-501 Advanced C++
- C++-502 Real-Time C++
- C-501 C for Real-Time Developers
- RTOS-201 Fundamentals of Real-Time Operating Systems

Course Workshop:

This course makes use of target hardware during the real-time practical exercises. The board targeted is an IAR Kickstart development board (NXP LPC2129 ARM7-based microcontroller). An application board is controlled via the LPC2129 to give attendees a real sense of embedded application development.

Course Outline:

Introduction to real-time systems
- What is a real-time and embedded computer system
- The need for a rigorous development procedure

From C to C++
- Non object-oriented C++ enhancements to basic C
- Conveniences of C++ over and above C

Introduction to Object Oriented (OO) Principles
- Key characteristics of OO development
- OO techniques and the real-time software development process

Introduction to Classes
- Classes & class instances
- Methods
- Constructors & destructors

More on Classes
- Inlining member functions
- const member functions
- static class members and functions
- arrays of classes
- implementing object relationships

Inheritance
- Building class hierarchies
- Dynamic binding for class methods, virtual functions
- Polymorphism

Multiple inheritance (MI)
- MI and interfaces

Functions and Operators
- Class defined conversions
- Overloading and function selection
- Friend functions
- Overloading operators
- Dynamic memory allocation revisited

Exception Handling
- What are exceptions?
- Throwing an exception
- The try block
- Catching an exception
- Rethrowing exceptions
- Catch all handlers
- Exception specifications

Templates
- Introduce parameterised types and functions
- Function templates
- Class templates

The Standard Library
- Introducing the Standard Template Library

Software Structuring
- Structuring large scale software systems
- Separate implementation from interface header files
- Dealing with name conflicts
- Linking with other languages

Embedded C++
- A summary of Embedded C++
- Embedded C++ features

Real-Time Specifics
- Low level facilities of C++ including:
  - Accessing hardware
  - Manipulating information at the bit level
  - Synchronising I/O with CPU via
  - Polling
  - Interrupts

Interrupt Programming
- Interrupt Service Routines in C++
  - functional approach
  - class approach

Target Specific Considerations:
- Data types;
- Language features affecting portability;
- Non-standard C++ language features;
- Assembly language interfacing;
- Designing ROMable objects.

Concurrency:
- Concurrency;
- Scheduling strategies;
- Sharing resources in multi-tasking systems;
- Synchronizing tasks;
- Transferring data between tasks.

Feabhas Ltd
5, Lowesden Works
Lambourn Woodlands
Hungerford, Berkshire
RG17 7RY, UK

Tel: +44 (0) 1488 73050
Fax: +44 (0) 1488 73051

Email:
info@feabhas.com
Web:
www.feabhas.com

Last update 07/06/2010