FEABHAS

Design Patterns in Modern C++

Course category	C++ Training Courses
Training area	Design Techniques
Course code	DP11-403
Duration	4 days
Additional information	Available for on-site delivery only. Can be delivered remotely or Face-to-Face.

Most software engineers can readily identify 'bad' code but they often lack the design skills to prevent such bad code happening in the first place. Languages like C++ are extremely expressive but, without a strong foundation in design it is easy to come up with fragile, inflexible, unmaintainable solutions.

Patterns provide flexible design solutions to commonly-occurring software problems.

This course is designed to provide delegates with a core understanding of design patterns and, more importantly, the principles and governors behind these patterns.

Students explore how these patterns can be implemented efficiently and effectively using Modern C++

It addresses the following topics:

- The need for intrinsic quality in software
- The principles of robust, flexible, maintainable software
- The difference between idioms, mechanisms and patterns
- The core design patterns
- How to think about, and apply, patterns to your code

Overview:

This 4 day course will provide practical, hands-on experience with the core design patterns, uniquely addressing their suitability to a non-PC programming environment.

Course objectives:

- Provide an understanding of the principles of object oriented design and how they relate to patterns.
- Provide a basic understanding of Design Patterns and how the language of patterns Provide practical experience of working with Design Patterns in Modern C++.
- Provide an understanding of the significant "Gang of Four" set of classical patterns and patterns

associated specifically with multi-tasking embedded systems.
- Gain the confidence to apply these new concepts to your next real-time project.

Pre-requisites:

- Good working knowledge of Modern C++
- An understanding of Object-Oriented principles
- UML modelling is useful, but not essential

Who should attend:

The course is aimed at software developers, designers, and architects wishing to improve their object-oriented design skills.

Duration:
Four days

Course materials:
Delegate handbook

Related courses:

- OO-504 Real-Time Software Design with UML
- C++11-501 Modern C++ for embedded systems
- C++11-502 Real-time Modern C++
- AC++11-401 Transitioning to C++11/C++14
- AC++11-401 Transitioning to C++11/C++14
- AC++11-501 Advanced Modern C++ for Embedded Systems
- TC++-401 Embedded Software Testing with C++

Course workshop:

The course exercises are designed to foster an understanding of design patterns, object orientation, and C++. Ample opportunity is provided for delegates to consider the implications of patterns to the size and space concerns of embedded systems whilst reflecting on the broader quality issues that they directly address.

Introduction to patterns

- Intrinsic quality
- Principles of modularisation
- Design patterns

Overview of OO Principles

- User-defined types (classes)
- Associsation
- Composition
- Specialisation
- Polymorphism
- Abstract base classes
- Interfaces
- Smart pointers

Construction patterns

- Singleton
- Factory patterns
- Abstract Factory
- Builder

Changing the interface

- Adapter
- Proxy
- Port
- Decorator

Structuring systems

- The pImpl idiom
- Bridge

Behavioural patterns
Observer
Command
State
Strategy

Real-time patterns

- Thread-Runs-Function
- Mutual exclusion
- Scope-locked idiom
- Monitor
- Guarded Suspension
- Asynchronous message
- Promise and future