C++ from C

An Introduction to C++ for C Programmers

ISO standard C++ is a general-purpose language that bridges different styles of programming and spans different platforms and different application styles.

The *C++ from C* course presents C++ by building from C and embracing modern C++ techniques. It develops the concepts and syntax through lectures, discussion and hands-on lab exercises.

Objectives

- Introduce C++ from a C perspective, appreciating some of its more advanced features
- Understand object-oriented and generic programming in C++
- Emphasise good practice and outline idioms for safe and sensible use of language features

Audience

The course is suitable for experienced C programmers. Previous knowledge of object-oriented concepts is advantageous but not required.

Content

A Brief Tour Differences from C · Characterising modern C++ · The role of objects and polymorphism · Classes · Templates · STL and the standard library

Simple Data Types Alternative initialisation and cast syntax · Pointer conversions · The *bool* type · Compile-time constants with *const* · Tags and type names

Convenient Library Types The library and *namespace std* · Using *string*, *vector* and *map* types

Functions Pass by reference and *const* reference · Guidelines on argument passing · Overloading and default arguments · Operator overloading · *inline* functions

Control Flow Declarations for loops and conditions · *try*, *throw* and *catch* · Exception safety

Dynamic Memory Management Type-safe allocation and deallocation with *new* and *delete*

Classes and Objects Encapsulation · Classes, member functions and data · *public* and *private* · *const* member functions · *class* versus *struct* · The *this* pointer

Object Relationships Delegation and forwarding · Composition and association

Construction and Destruction Constructors and destructors · Default constructors · Member initialiser lists · Construction and destruction order

Value Objects Conversions for value objects · *explicit* · Default copying · Copy constructors · Member and non-member operators · Copy assignment · Copy prevention for non-value types

Exception Objects Using objects for exceptions · Safe copy assignment · Safe resource release

Templates Generic programming · Function templates · Class templates · Templates in the standard library · *auto_ptr* for exception safety · Managing long type names

Containers The standard sequence containers · The standard associative containers

Iterators Iterator categories · Container iterators · *iterator* and *const_iterator* · Stream iterators

Algorithms Sorting, searching and copying ranges · Refactoring loops with generic algorithms

Function Objects Using function objects with algorithms · Adapting associative containers

Interface-Based Programming Interface classes and pure *virtual* functions · *virtual* destructors · Implementing interfaces with *public* derivation and overridden functions · Multiple base classes

Inheritance-Based Programming Generalisation, specialisation and substitutability · Abstract classes with some implementation · *protected* · Base class construction · RTTI · Exceptions

Additional Details

Duration 4 days

Setup Projection facilities for a laptop · One workstation per delegate with C++ compiler installed (configuration to be agreed) · Whiteboards and/or flip charts

Contact Kevlin Henney · kevlin@curbralan.com · Curbralan Limited · +44 117 942 2990