Object Programming in C++

Hands-on course of 5 days - 35h
Ref.: C++ - Price 2024: CHF2 970 (excl. taxes)

EDUCATIONAL OBJECTIVES

At the end of the training, the trainee will be able to:

Master the syntax of the C++ language

Implement the concepts of Object-Oriented Design

Use development tools associated with the language C++

Master major additions from the C++ 11 standard

Provide workstations equipped with Visual C++ (in Windows) and gcc (in Unix). The HOW was designed to illustrate all asp

Instructional methods

All exercises include an analysis/design phase following by a programming phase.

TRAINING PROGRAM

TEACHING METHODS

All the exercises include an analysis/design phase followed by a programming phase.

Workstations with the Visual C++ (in Windows) and gcc (in Unix) languages are provided. The hands-on work was designed to illustrate all aspects of the language and to always implement object-oriented design concepts.

THE PROGRAMME

last updated: 01/2018

1) C++ syntax (differences between C and C++)
- Data: Definition, initialization, types of data.
- Expressions: Notion of reference, casting mechanisms.
- Operators (: :, new, delete).
- Functions (passing parameters and returning values by reference, default values, inlining, overload).
- Using C code in a C++ program.
- References (arguments and return values).
- Constant types.
- Namespaces.
- "Automatic" typing with the keyword auto (C++ 11).
- Hands-on work ¤Getting started with the development environment and programming a simple program.
Getting started in the development environment and coding a simple program.

2) Object-Oriented Approach
- General principles of Object techniques.
- C++ and Object programming.
- Introduction to Object-oriented methodologies.
- An introduction to UML diagrams and rating (static diagram, dynamic diagram, collaboration diagram, scenario).
- Hands-on work ¤Applying concepts to a case study that will be central to the exercises that follow.
Applying the concepts to a case study, which will be one of the recurring themes in the exercises that follow.

3) C++ classes and objects.

- Syntactical aspects: Fields, methods, constructors.
- Access control.
- Self-reference.
- Fields and static methods.
- Functions.
- Friend methods and classes.
- Dynamically creating object tables.
- Methodological aspects: Designing classes.
- Copy and move constructors (C++ 11).
- Delegating constructors (C++ 11).
- Introduction to memory management issues (stack, heap, garbage collector).
- Hands-on work ¤Programming the case study. Designing and building a class and interface hierarchy.
Programming the case study. Designing and constructing a class and interface hierarchy.

4) Derivation and inheritance.

- Principle of derivation.
- Syntactical aspects: Definition of derived classes, constructors.
- Access control.
- Implementing polymorphism: Virtual functions.
- Reusing code: Abstract classes.
- Interfaces.
- Multiple inheritance.
- Semantic and methodological aspects: Code factoring.
- Hands-on work ¤Setting up polymorphism in the case study.
Setting up polymorphism in the case study.

5) Exceptions

- Syntactical aspects: Try blocks, generating exceptions.
- Methodological aspects: Constructing an exception hierarchy, use of exceptions.
- Hands-on work ¤Introducing exceptions into the case study.
Introduction of exceptions in the case study.

6) Overloading operators

- Principle of overload.
- Overloading binary operators.
- Particular overload: Index operator, function, conversion.
- Overloading memory management operators.
- Overloading the operators '<<' and '>>'.
- Hands-on work ¤Overloading some simple operators.
Overloading several simple operators.

7) Models

- Class models. Principles and general mechanisms. Overloading models and redefining methods.
- Function model. Principles and general mechanisms. Overloading templates.
- Templates and overloading operators.
- Templates and derivation mechanisms.
- Improvements offered by C++ 11.
- Hands-on work ¤Model exercises.
Exercises in templates.

8) I/O and overview of STL

- I/O.
- The principle of streams and input/output class hierarchy.
- Description of some input/output classes.

- Overview of STL.
- Objectives and principles.
- Descriptions of some templates and classes.
- Containers, iterators, range-based for loop (C++ 11).

9) Conclusion

- Software lifecycle: Testing, integration, release method.
- Interaction with other environments.
- Critical analysis of C++.
- Evolution of C++.

DATES

REMOTE CLASS
2024 : 23 Sep, 02 Dec