Ambient IT
128 Boulevard Macdonald
75019 Paris

Mail : formation@ambient-it.net
Tél : +33 (0)1 40 34 11 53

AMBIENT-IT

C# Language Training 8.0

Presentation

It is undeniably one of the most versatile programming languages available for engineers today.

It is an object-oriented programming language, marketed by Microsoft since 2002 and intended for to be developed on the Microsoft.NET platform.

It is derived from C++ and is very close to Java, from which it uses the general syntax as well as the concepts, including

adding concepts such as operator overload, indexers and delegates. It is used notably to develop web applications on the ASP.NET platform.

Objectives

- Know the latest pattern implementations available in C# and F#.
- Refer to tried and tested pattern variations.
- Study comprehensive, stand-alone examples, including many examples that cover advanced scenarios.
- Use the latest implementations of C# and Visual Studio/ReSharper 8.0

Target audience

Developers, architects, technical project managers.

Prerequisites

- Working knowledge of the C# language
- Development with the .NET Framework.

Program

introduction

- C# definition features
- Managed Code and CLR
- Prefer generality to specialization
- Visual Studio and Visual Studio Code
- Anatomy of a simple program
- Adding a project to an existing solution
- Reference to a project from another project
- Referencing external libraries
- Writing a unit test
- Naming spaces
- Classes
- Program entry point
- SOLID design principles
- The functional perspective

Exploration of c#

- Using and implementing abstract classes
- Use and implementation of interfaces
- Asynchronous programming using asynchronous and standby
- Use of extension methods
- Generics
- Null type
- Dynamic type

New features in C# 8.0

- Nullable reference types
- Recursive patterns
- Ranges and indices
- Change expression
- New expressions typed by target
- Asynchronous flux
- Use of statements
- Patterns positional
- Patterns tuples

Data structures and optimized code writing in C#.

- Understand the use of Big O notation to measure the performance and complexity of an algorithm
    - Logarithms
- Choosing the right data structure to optimize performance
        - Tables
        - Lists
        - Batteries
        - Waiting queue
        - Related lists such as Individually linked lists, Double-linked lists or Related Circular Lists
- Dictionaries, hash tables and hashsets
- Generic lists
- Better practices in writing optimized code in C#.
    - Boxing and unpacking overheads
    - String concatenation
    - Exception handling
    - For and foreach
    - Delegates

Write, debug and test functions

- Writing functions
        - Writing a time table function
        - Write a function that returns a value
        - Write mathematical functions
        - Formatting issues for publishing
        - Factorial calculation with recursivity
- Debugging an application under development
    - Create an application with a deliberate bug
    - Setting a breakpoint
    - The debugging toolbar
    - Debugging windows
    - Going through the code
    - Customization of breakpoints
- Registration during development and execution
    - Instrumentation with debugging and tracing
        - Write to the default trace earpiece
        - Configuration of trace auditor
    - Switching trace levels

Control of flow and conversion types

- Declarations of selection
    - Using Visual Studio 2017
    - The if instruction
        - The code
        - Pattern matching with the if statement
    - The switching command
        - The code
        - Pattern matching with the switching instruction
    - Statements on iteration
    - The declaration of while
    - The instruction do
    - The instruction for
    - The foreach statement
- Casting and conversion between types
    - Casting of numberss
        - Implicit casting of numbers
        - Casting numbers explicitly
    - Using the conversion type
    - Rounded figures
    - Conversion of any type into string
    - Converting a binary object into a string
    - String analysis for numbers or dates and times
- Exception handling during type conversion
    - The test declaration
    - All exceptions taken into account
    - Specific exception handling
- Overflow check
    - The audited statement
    - The unverified statemente

Creative designs

- Builder
- Plants
- Prototype
- Singleton

Structural models

- Adapter
- Bridge
- Composite
- Decorator
- Facade
- Flyweight
- Power of Attorney

Behavioural Models

- Chain of custody
- Order
- Interpreter
- Iterator
- Mediator
- Memento
- Null subject
- Observer
- Status
- Strategy
- Template method
- Visitor

Companies concerned

This training is intended for companies, small or large, wishing to train their teams in a new advanced computer technology.

Teaching methods

Practical Training: 60% Practical, 40% Theory. Training support distributed in digital format to all participants.

Organization

The course alternates the trainer's theoretical inputs supported by examples and sessions of reflections, and group work.

Validation

At the end of the session, a multiple-choice questionnaire is used to check the correct acquisition of the skills.

Sanction

A certificate will be given to each trainee who has completed the entire training.