Over 35 Years Of Technology Training

Document Generated: 04/06/2026

Learning Style: Virtual Classroom

Technology:

Difficulty: Intermediate

Course Duration: 5 Days

Next Course Date: May 11, 2026

Intermediate C# | Patterns, Generics, Threading, Database, Entity Framework, UI & More (TTCN2155)

About This Course:

Developing Applications in C# and .Net Core (Intermediate C# / .Net Core) is a lab-intensive, hands-on course designed to provide you with the skills required to take your C# programming skills to the next level. Throughout the course you'll explore C# Design Guidelines, Robust Classes, Generics, Lambda Expressions & LINQ, Reflection & Metadata, Resources & Globalization, Basics of Threading and the Basics of Database connectivity.

Course Objectives:

- Design and implement classes that adhere to SOLID design principles
- Understand common design patterns
- Implement robust classes with properties, methods, delegates & events
- Understand, use, and implement generics
- Explore generics in the .Net Framework
- Understand and use Language integrated Query (LiNQ)
- Understand and use Lambda expressions
- Build dynamic applications using reflection
- Use internationalization to build world friendly applications
- Build multithreaded applications
- Understand the basics of database access
- What's new in .Net Core and C#?

Audience:

- This is an intermediate-level .Net programming course, designed for experienced C# developers

Prerequisites:

- TTCN20483 Programming in C# / Creating Apps in C# and .Net Core

Course Outline:

Overview of new Features

- Type specifiers

- Raw Literals
- Top-level statements
- Records
- Global Usings
- Tuples
- Deconstruction

Design Guidelines

- Conventions
- Class Design guidelines
- Exception Design guidelines
- Common Design Patterns

Robust Classes

- Properties
- Functions
- Delegates
- Events

Unit Testing with XUnit

- Utilizing Test Driven Development (TDD)
- Survey of Unit test frameworks
- Using XUnit
- Mocking with Moq

Exceptions

- Overview

- Syntax
- Exceptions vs Return code
- Testing for exceptions

Generics

- Generic Classes
- Generic Functions
- Understanding invariance
- Survey of Generics in the .Net Library

Delegates

- Overview
- Methods as objects
- Generics and Delegates
- Delegates in the library

Events

- Overview
- Defining
- Register/Deregister

Lambda Expressions

- Using lambdas to generate a delegate

Inheritance

- Overview
- Syntax
- Override/Virtual

- Abstract
- Polymorphism

SOLID Principles & Design Patterns

- Overview
- Design Smells
- Single Responsibility
- Open/Close
- Liskov's Substitution Principle
- Using SOLID to correct design smells

Interfaces

- Overview
- Interface motivations and SOLID
- Interface Segregation
- Dependency Inversion
- Default interface methods

LInQ

- Overview
- Extension Methods
- Linq API
- Linq Query

Attributes, Reflection & Metadata

- Reflections basics
- Attributes

Resources & Globalization

- Overview
- Resource Bundles
- Specifying Language/Region

Threads, Tasks, Async/Await

- Creating Threads
- Coordinating threads
- Tasks
- Async/Await

Basics of Entity Framework

- DbContext
- Code first vs Db first
- Entities
- Mapping

Survey of GUI Applications

- MAUI
- Blazor
- Asp.Net