Advance Programming Techniques (APT)

Lecture # 1

Ehtisham Rasheed

Department of Computer Science
University of Gurjat, Gujrat

UNIVERSITY OF GUJRAT

Advanced Programming Techniques using C#

- Prerequisite
  - Programming Fundamentals
  - Object Oriented Programming

What is .NET?

- **.NET** is a free, open source, secure, cross-platform framework developed by Microsoft.
- It provides:
  - **Languages** C#, F#, VB.NET
  - **Runtime** (Common Language Runtime, CLR)
  - **Class libraries** (pre-built functions for file handling, networking, databases etc)

Common Language Runtime (CLR)

- At the core of .NET ecosystem is the Common Language Runtime (CLR).
- Responsible for code execution written in different .NET languages
- Provides services
  - Memory management
  - Garbage collection
  - Security
- Code in C#, F#. VB.NET and other .NET languages compiled into Intermediate Language (IL) that the CLR can execute

Evolution of .NET

- **.NET Framework**
  - Released in **2002** by Microsoft
  - Runs **only on Windows**
  - Used for **desktop apps** (WinForms, WPF), **ASP.NET Web Forms**, **enterprise apps**
  - Latest and final version is **.NET Framework 4.8 (2019)**
  - Microsoft still provides **security fixes**, but no new features

Evolution of .NET

- **.NET Core**
  - Released in **2016** as new **lightweight** and **cross-platform** version
  - Runs on **Windows**, **macOS**, **Linux**
  - Modern, open-source, high performance
  - Supports **cloud**, **web APIs**, **microservices**
  - **Side-by-side** versioning (multiple versions can coexist)
  - Replaced the old **.NET Framework** for new projects

Evolution of .NET

- **.NET (5, 6, 7, 8, …)**
  - **Starting from .NET 5 (2020)**, Microsoft merged .NET Framework and .NET Core into a single platform called **".NET"**
  - So now when we say “.NET,” we mean the **modern version** (not the old Framework)
  - Supports **desktop, web, mobile, cloud, AI, IoT**
  - Cross-platform, open-source, continuously updated
  - **Long-Term Support (LTS) releases** (e.g., .NET 6, .NET 8)

Why Learn C#?

- C# is a general-purpose language used for a broad range of applications, including:
  - Web Development
  - Desktop Applications
  - Game Development
  - Mobile Development
  - Cloud and Enterprise Applications
  - AI and IoT
  - Microsesrvices

IDE for C#

- Visual Studio
  - Developed by Microsoft, most common IDE for C#
  - It has lot of features like IntelliSense, Advanced Debugging Tools, Integrated Git Support etc.
  - Free community edition for individual developers
  - Paid version is also available

IDE for C#

- Visual Studio Code (VS Code)
  - Also developed by Microsoft
  - Light-weight, cross-platform editor for C# support through extensions
- JetBrains Rider
  - A powerful and feature-rich cross-platform .NET IDE developed by JetBrains
  - Rider is known for its intelligent code analysis, refactoring capabilities, and excellent performance
  - Particularly used by developers working on macOS or Linux

C# Compilation Model

- C# Source Code → Compiler → Intermediate Language (IL / MSIL)
- MSIL → CLR → JIT Compiler → Native Machine Code
  - CLR converts MSIL to machine code at runtime
  - This allows cross-platform support, as each OS has its own JIT compiler
- CLR manages execution (memory, security, garbage collection)
- Any .NET language (C#, VB.NET, F#) compiles to IL, so they can interpolate

Frist Program

- Learn how to create new Console Application in C#
- Learn how to create new Windows Application in C#

Desktop Application Development

- Software that runs locally on a computer (not in the browser)
- Examples: Notepad, Paint, MS Word, Calculator
- Built using frameworks like WinForms, WPF, WinUI, JavaFX etc

Evolution of Microsoft Desktop Frameworks

Framework	Year Introduced	Features	Current Status
WinForms	2002	Easy drag-and-drop, event-driven	Still supported but legacy
WPF	2006	XAML, MVVM, animations, 2D/3D graphics	Supported, less focus
Win UI 3	2021	Fluent design, modern controls	Microsoft’s current focus
.NET MAUI	2022	Cross-platform (Windows, macOS, iOS, Android)	Future direction

Why Start with WinForms?

- Easier to learn -> drag & drop interface
- Fast prototyping for beginners
- Focus on logic, events, OOP instead of UI complexity (XAML, MVVM)
- Still widely used in enterprise apps
- Massive ecosystem & resources
- Still relevant in the job market