Report on .NET Versions, Namespace, .NET Core, and Solution in C#

I. .NET versions:

.NET is a free, open source platform that provides a unified framework for various types of applications that enables developers to build and run applications for different operating systems.

There are 3 different versions of .NET with their updates every year. Let's talk in more details about them.

1. .NET Framework:
    - It was first released at 2002 as version 1.0 .
    - The last stable version is 4.8.x which was released at 2024.
    - It's the first platform designed for windows-based applications.
    - It includes a base class library, the Common Language Runtime (CLR), and it supports various application types such as desktop, web (ASP.NET), and services.
    - Base class library provides libraries that support basic functions.
    - The Common Language Runtime (CLR) which handles memory management, garbage collection, exception handling, and other core system tasks.
    - ASP.NET: This is the framework for building dynamic web applications. Now it's evolved into ASP.NET MVC.
    - Windows Forms and WPF (Windows Presentation Foundation): they were used to create desktop applications with GUI support on Windows.
    - It has many limitations like : it only supports windows which allows developers to only develop windows desktop apps and web applications. Also It enforces developer to download all libraries in the framework which decreases modularity and Increases runtime.

In addition to that, It isn't cross-platform, which means that developer can't make an applications for all operating systems.

2. .NET Core:
    - It was first released in 2016 with version 1.0 .
    - When it reached version 3.0, It became a strong alternative for .NET Framework as it solved many problems of it like not to be a cross-platform and the much runtime.
    - It's cross platform.
    - It increased execution speed as it doesn't force the developer to download all unnecessary libraries
    - It's open source.
    - It provides modularity that any developer can install only the parts of the framework they need via NuGet packages.
    - CLI Tools: .NET Core introduced a set of command-line interface (CLI) tools, allowing developers to create, build, test, and publish applications using just the command line.
    - With the introduction of .NET 5 in November 2020, Microsoft unified .NET Framework and .NET Core into a single platform.
3. .NET 5 and after:
    - It was released in November 2020, unifying .NET Framework and .NET Core into a single platform under .NET name and is made for all types of applications and operating systems.
    - It's all-in-one, you can develope any application in the same platform. So it's also a cross-platform.
    - It enhances performance with many techniques like enhanced Just-In-Time (JIT) compilation and by doing improvements to garbage collection.
    - It provided support like pattern matching.
    - It uses minimal APIs making it simpler and faster for developers to get started.

- It has many versions like: .NET 6 in 2021, .NET 7 in 2022, .NET8 in 2023 and finally .NET 9 from about 2 weeks only !!

II. Namespaces in .NET

1. Role:
    - It's a way of grouping related classes logically.
    - It helps to avoid conflicts in naming and organizes the code in a more readable way.
2. Usage:
    - It's used to avoid name. Many classes may have the same name but namespace ensures that not any of these classes are conflicted with one other.
    - It helps to organize the code by putting the related type together.
    - It helps to divide the code into smaller modules which achieves modularity.
    - The common namespace is System which is the root of all namespaces in .NET Core.

III. .NET Core

1. Foundation:
    - It's considered the new phase of .NET Framework as it provides the same services (providing unified framework for various types of applications) but with application of cross-platform to be operable in any operating system.
    - It was first released in 2016 with version 1.0 .
    - When it reached version 3.0 at 2018, It became a strong alternative for .NET Framework.
2. Features:
    - It's modular and lean (by including any library the developer needs).

- It provides better performance than ASP.NET as it is ASP.NET Core.
- It's a unified development environment where developer can make any application for any operating system.

IV. Solutions

1. Definition
    - It's a container that manages multiple related projects.
    - It's a way to group together various projects within a single logical unit.
    - These projects can be: (Web Applications, Class Libraries, Unit Tests, DB Projects, Desktop Applications and Mobile Applications).
    - Each project is independent from the other, but they work together to complete system.
    - The Solution file (.sln) is the core file of a solution and it is used to organize and manage the projects. It contains references to the projects in the solution.
2. Benefits:
    - Modularity.
    - Independence: Each project in same solution is tested independently.
    - Version Control: If the solution is unified structure, it will make it easy to be managed using version control like git.
3. Configuration:
    - Solution file not only references projects but can also define how the projects are built.
    - These configurations could be: (Build Configuration like: Debug),(Platform Targeting like: x86 or x64),(NuGet Package Restore: as when the solution file restores these packages automatically when they are opened in visual studio),(Solution-Specific Build Order: In solutions with multiple projects, the build order is

maintained to ensure dependencies are built in the correct order).

- It’s managed by selecting build then Configuration Manager to choose the configuration for each type.