**Architecture of .Net framework**

**C# (C-Sharp)** is a modern, object-oriented programming language developed by Microsoft in 2000. It is a part of the .NET ecosystem and is widely used for building desktop, web, mobile, cloud, and enterprise applications. This is originally tied to the .NET Framework, C# has evolved to be the primary language used across the .NET platform which now supports cross-platform development.

The **.NET Framework** initially launched in 2002 and it has evolved into a broader and more powerful ecosystem known as .NET 5 and beyond. This platform combines the best of .NET Framework, .NET Core, and Xamarin and it enables the developers to build applications that run on Windows, Linux, and macOS.

**Key Components of the .NET Framework**

The .NET Framework’s basic architecture consists of two key elements:

1. **Common Language Runtime (CLR)**: The Common Language Runtime (CLR) is the core runtime engine of the .NET Framework, responsible for managing the execution of code written in any of the .NET-supported languages, such as C#, F#, and VB.NET. It provides various services, including:

* Memory Management: Automatic memory allocation and garbage collection.
* Code Safety: Ensures type safety and handles security through Code Access Security (CAS).
* Execution Management: Converts Intermediate Language (IL) code into machine code via the Just-In-Time (JIT) compiler.

The CLR enables Cross-Language Interoperability, which allows components written in different languages to work together seamlessly.

**2. .NET Framework Class Library (FCL)**: The .NET Framework Class Library (FCL) is a vast collection of pre-built functions and classes available to developers. The FCL provides a range of namespaces that contain classes for tasks like:

* File I/O
* Networking
* Database access
* Graphical User Interface (GUI) design

Additionally, it includes tools such as the Visual Studio IDE, which simplifies application development, debugging, and testing.

**3. Other Key Components:**

* **Common Type System (CTS)**: It ensures that types are consistent across different languages within the .NET ecosystem.
* **Common Intermediate Language (CIL):**This is a low-level language that is compiled from source code and later converted into machine code by the JIT compiler.
* **Assemblies**: The bundles of code that are versioned and deployed together, providing the essential building blocks of a .NET application.

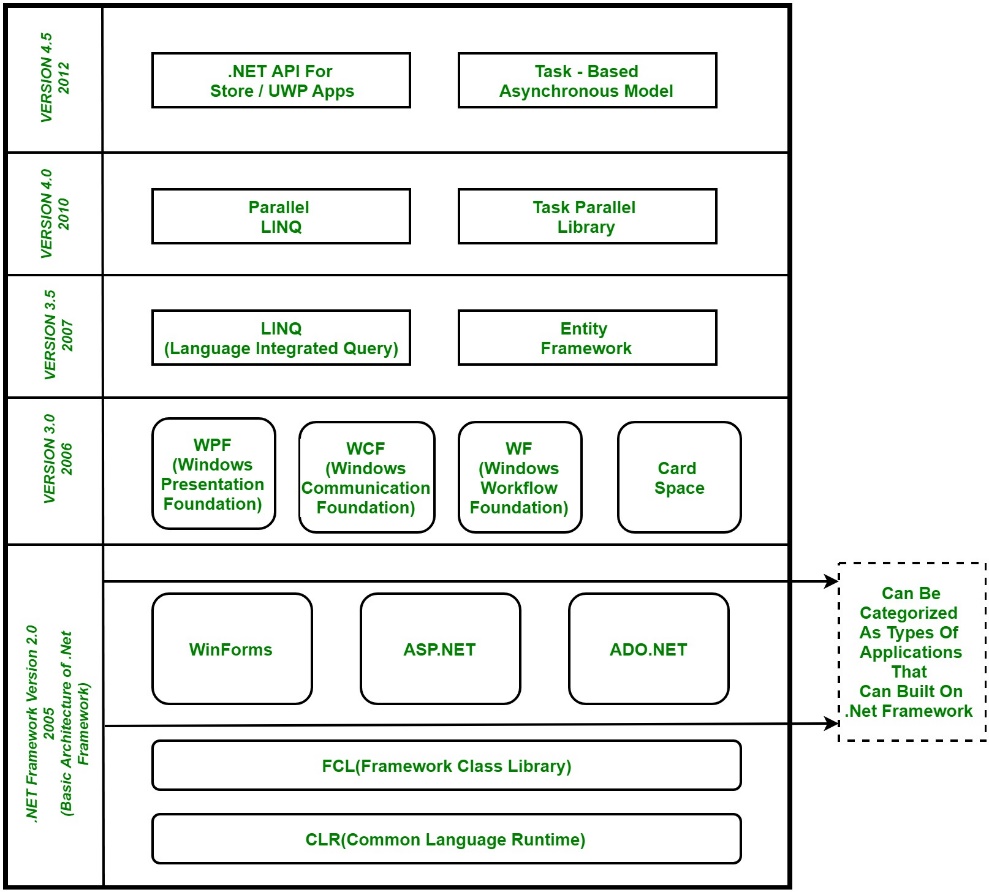
**Evolution of .NET Framework to .NET 5+**

Initially the .NET Framework was designed to be Windows-specific but with the introduction of .NET Core (starting from version 1.0) in 2016, Microsoft cross-platform development. .NET Core allowed developers to create applications that could run on Windows, Linux, and macOS.

With the release of .NET 5 in 2020, Microsoft unified .NET Framework, .NET Core, and Xamarin into a single platform, making .NET 5+ the modern, cross-platform successor. This new platform focuses on performance improvements, security, and a rich set of libraries that can be used for various types of applications.

**Basic Architecture and Component Stack of .NET Framework**

The first three components from the bottom are considered as the basic architecture of .Net framework which came in the year 2005 and after this more components were added by Microsoft in the .Net Framework as following:

[](https://media.geeksforgeeks.org/wp-content/uploads/Net-Framework.jpg)

**Note:** The diagram illustrates the architecture of the **.NET Framework from version 2.0** (released in 2005) through **version 4.5 (released in 2012)**. But it is important to note that since the introduction of **.NET Core in 2016** and the unification into **.NET 5+ starting in 2020**, the architecture has continued to evolve, with the latest versions (such as **.NET 6, .NET 7, and .NET 8**) offering enhanced performance, cross-platform support, and a unified development experience. These modern versions of .NET are designed to run on Windows, Linux, and macOS, further advancing the ecosystem in 2025 and beyond.

1. **CLR (Common Language Runtime):** It is a run-time environment that executes the code written in any .NET programming language. .Net framework provides support for many languages like C#, F#, C++, Cobra, Jscript.Net, VB.Net, Oxygene, etc.
2. **FCL (Framework Class Library) :** A large number of class libraries are present in this framework which is known as FCL.
3. **Types of Applications:** Mainly the applications that are built in the .Net framework are divided into the following three categories :
   1. **WinForms:** Form–Based applications are considered under this category. In simple terms, we can say client-based applications that read and write the file system come under this category.
   2. **ASP .NET:** Web-based applications come under this category. ASP.Net is a framework for the web and it provides an awesome integration of HTML, CSS, and JavaScript which makes it useful to develop web applications, websites, and web services. **Web services were added in .Net Framework 2.0 and considered as a part of ASP.NET web applications.**
   3. **ADO .NET:** It includes the applications that are developed to communicate with the database like MS SQL Server, Oracle, etc. It mainly consists of classes that can be used to connect, retrieve, insert, and delete data.
4. **WPF (Windows Presentation Foundation):** Windows Presentation Foundation (WPF) is a graphical subsystem given by Microsoft that uses DirectX and is used in Windows-based applications for rendering UI (User Interface). WPF was initially released as part of .NET Framework 3.0 in 2006 and was previously known as **“Avalon”**.
5. **WCF (Windows Communication Foundation):** It is a framework for building connected and service-oriented applications used to transmit data asynchronously from one service endpoint to another service point. It was previously known as the **Indigo**.
6. **WF (Windows Workflow Foundation):**It is a technology given by Microsoft that provides a platform for building workflows within .Net applications.
7. **Card Space:**It is a Microsoft .NET Framework software client that is designed to let users provide their digital identity to online services in a secure, simple, and trusted way.
8. **LINQ (Language Integrated Query):** It is introduced in .Net framework version 3.5. Basically, it is a query language used to make the query for data sources with VB or C# programming languages.
9. **Entity Framework:** It is an open–source ORM (Object Relational Mapping) based framework that comes into .Net Framework version 3.5. It enables the .Net developer to work with a database using .Net objects. Before the entity framework, .Net developers performed a lot of things related to databases. Like to open a connection to the database, developers have to create a Data Set to fetch or submit the data to the database, convert data from the Data Set to .NET objects, or vice-versa. It creates difficulties for developers and also it was an error-prone process, then **“Entity Framework”** comes to automate all these database-related activities for the application. So, Entity Framework allows the developers to work at a higher level of abstraction.

### Parallel Programming and Performance Enhancements

1. **Parallel LINQ (Language Integrated Query):** It comes in .Net Framework version 4.0 and is also termed PLINQ. It provides a concurrent query execution engine for **LINQ**. It executes the **LINQ** in parallel such that it tries to use as much processing power the system on which it is executing.
2. **TPL (Task Parallel Library):** It is a set of public types and APIs. It allows the developers to be more productive by simplifying the process of adding concurrency and parallelism to .Net applications.
3. **.NET API For Store/UWP Apps:** In 2012, Microsoft added some APIs for creating **UWP(Universal Windows Platform)** apps for Windows using C# or VB.
4. **Task-Based Asynchronous Model:** It is a model used to describe the asynchronous operations and tasks in the .Net Framework.