Learn C# in 1 Day

By Krishna Rungta

Table Of Content

Chapter 1: What is .NET Framework? Complete Architecture Tutorial

1. What is Microsoft .Net Framework?
2. .Net Framework Architecture
3. .NET Components
4. .Net Framework Design Principle

Chapter 2: C# and .Net Version History

1. .Net Framework Version History
2. C# Version History

Chapter 3: How to Download and Install Visual Studio for C#

1. How to Download and Install Visual Studio
2. Visual Studio Key Features

Chapter 4: C# Hello World: First Console Application Program

Chapter 5: C# Data Types with Example

Chapter 6: C# Enum(Enumeration) with Example

Chapter 7: C# Variables & Operators with Example

1. C# Variables
2. C# Operators

Chapter 8: C# IF, Switch, For, While Loop Statements Tutorial [Examples]

1. Flow Control and conditional statements
2. If statement
3. Switch statement
4. While loop
5. For loop

Chapter 9: C# Array Tutorial: Create, Declare, Initialize

Chapter 10: C# Class & Object Tutorial with Examples

1. What is Class and Object?
2. How to Create a Class and Object
3. Fields and methods

Chapter 11: C# Access Modifiers(Specifiers) with Example

1. Access Modifiers
2. C# Constructor

Chapter 12: C# Inheritance & Polymorphism with Examples

1. What is Inheritance in C#?
2. What is Polymorphism in C#?

Chapter 13: C# Abstract Classes Tutorial with Example

Chapter 14: C# Interface Tutorial with Example

Chapter 15: C# Collections Tutorial with Examples

Chapter 16: C# ArrayList Tutorial with Examples

Chapter 17: C# Stack with Example

Chapter 18: C# Queue with Examples

1. What is Queue in C#?
2. C# Queue Dequeue

Chapter 19: C# Hashtable with Examples

Chapter 20: C# Windows Forms Application Tutorial with Example

1. Windows Forms Basics
2. C# Hello World
3. Adding Controls to a form
4. C# Event Handling for Controls
5. Tree and PictureBox Control

Chapter 21: C# Database Connection Tutorial with Example

1. Fundamentals of Database connectivity
2. How to connect C# to Database
3. Access data with the SqlDataReader
4. C# Insert Into Database
5. C# Update Database

6. Deleting Records
7. Connecting Controls to Data
8. C# DataGridView

Chapter 22: C# File I/O Handling Operations [Examples]

1. Basics I/O Commands
2. File.Exists
3. File.ReadAlllines
4. File.ReadAllText
5. File.Copy
6. File.Delete

Chapter 23: C# Stream Tutorial: StreamReader, StreamWriter with Example

1. Stream Reader
2. Stream Writer

Chapter 24: C# Serialization & Deserialization with Example

Chapter 25: Coded UI Test Automation Framework Tutorial

Chapter 1: What is .NET Framework? Complete Architecture Tutorial

What is Microsoft .Net Framework?

The .Net framework is a software development platform developed by Microsoft. The framework was meant to create applications, which would run on the Windows Platform. The first version of the .Net framework was released in the year 2002.

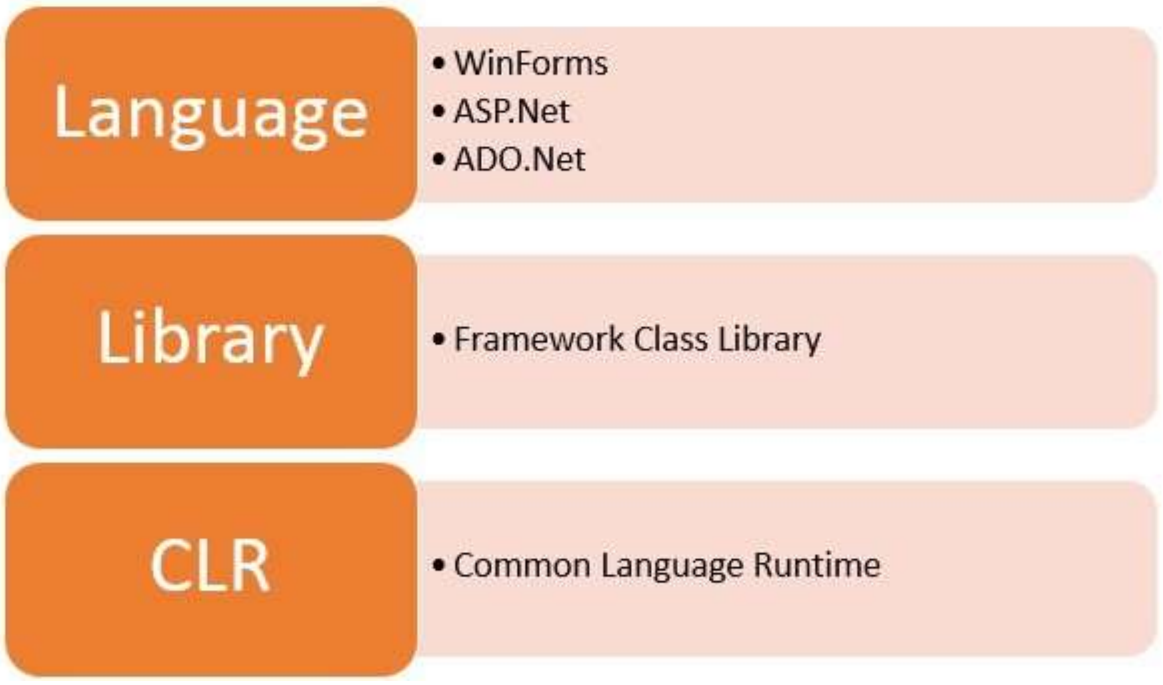
The version was called .Net framework 1.0. The .Net framework has come a long way since then, and the current version is 4.7.1.

The .Net framework can be used to create both - **Form-based** and **Web-based** applications. Web services can also be developed using the .Net framework.

The framework also supports various programming languages such as Visual Basic and C#. So developers can choose and select the language to develop the required application. In this chapter, you will learn some basics of the .Net framework.

.Net Framework Architecture

The basic architecture of the .Net framework is as shown below.



.net framework architecture diagram

.NET Components

The architecture of the .Net framework is based on the following key components;

1. Common Language Runtime

The "Common Language Infrastructure" or CLI is a platform on which the .Net programs are executed.

The CLI has the following key features:

- Exception Handling - Exceptions are errors which occur when the application is executed.

  Examples of exceptions are:

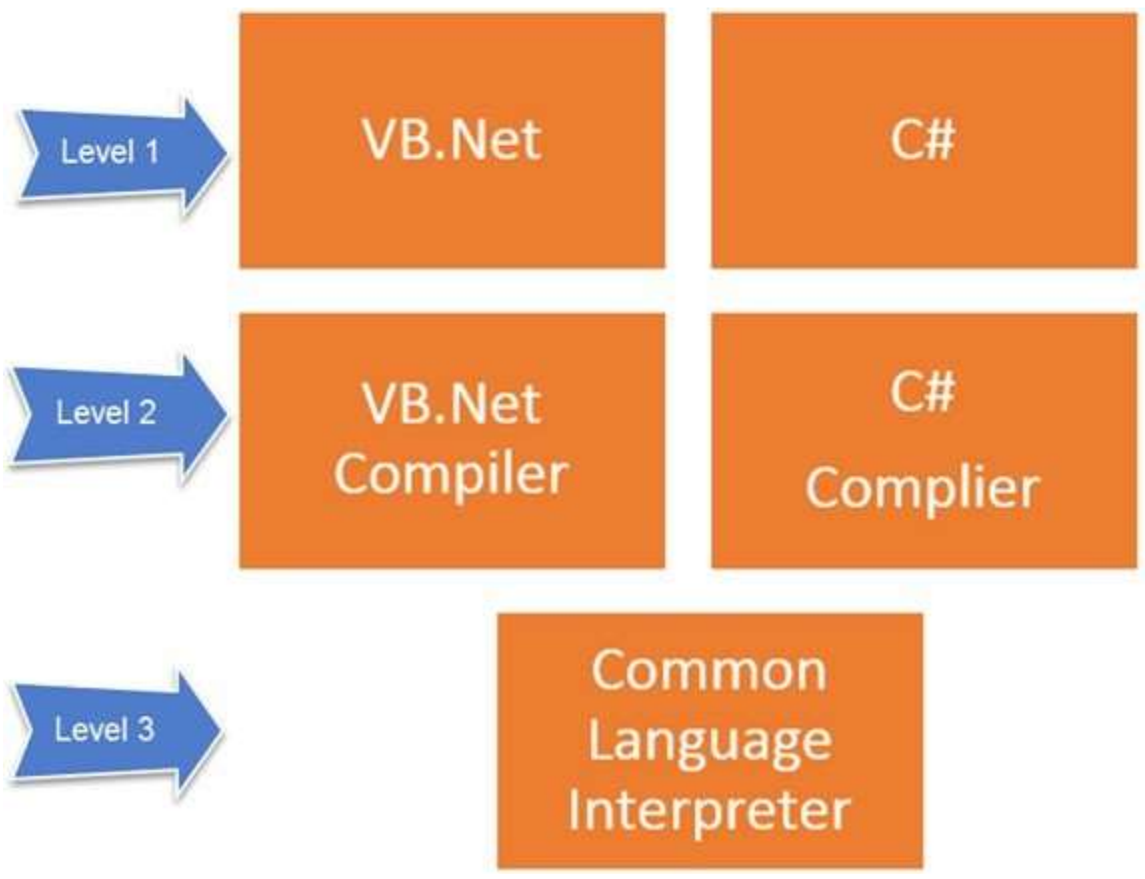
    - If an application tries to open a file on the local machine, but the file is not present.
    - If the application tries to fetch some records from a database, but the connection to the database is not valid.
- Garbage Collection - Garbage collection is the process of removing unwanted resources when they are no longer required.

  Examples of garbage collection are

    - A File handle which is no longer required. If the application has finished all operations on a file, then the file handle may no longer be required.
    - The database connection is no longer required. If the application has finished all operations on a database, then the database connection may no longer be required.
- Working with Various programming languages –

As noted in an earlier section, a developer can develop an application in a variety of .Net programming languages.

1. Language - The first level is the programming language itself, the most common ones are VB.Net and C#.
2. Compiler – There is a compiler which will be separate for each programming language. So underlying the VB.Net language, there will be a separate VB.Net compiler. Similarly, for C#, you will have another compiler.
3. Common Language Interpreter – This is the final layer in .Net which would be used to run a .net program developed in any programming language. So the subsequent compiler will send the program to the CLI layer to run the .Net application.



2. Class Library

The .NET Framework includes a set of standard class libraries. A class library is a collection of methods and functions that can be used for the core purpose.

For example, there is a class library with methods to handle all file- level operations. So there is a method which can be used to read the text from a file. Similarly, there is a method to write text to a file.

Most of the methods are split into either the System.* or Microsoft.* namespaces. (The asterisk * just means a reference to all of the methods that fall under the System or Microsoft namespace)

A namespace is a logical separation of methods. We will learn these namespaces more in detail in the subsequent chapters.

3. Languages

The types of applications that can be built in the .Net framework is classified broadly into the following categories.

- WinForms – This is used for developing Forms-based applications, which would run on an end user machine. Notepad is an example of a client-based application.
- ASP.Net – This is used for developing web-based applications, which are made to run on any browser such as Internet Explorer, Chrome or Firefox.
  - The Web application would be processed on a server, which would have Internet Information Services Installed.
  - Internet Information Services or IIS is a Microsoft component which is used to execute an Asp.Net application. The result of the
  - execution is then sent to the client machines, and the output is shown in the browser.
- ADO.Net – This technology is used to develop applications to interact with Databases such as Oracle or Microsoft SQL Server.

Microsoft always ensures that .Net frameworks are in compliance with all the supported Windows operating systems.

.Net Framework Design Principle

The following design principles of the .Net framework is what makes it very relevant to create .Net based applications.

1. Interoperability - The .Net framework provides a lot of backward

support. Suppose if you had an application built on an older version of the .Net framework, say 2.0. And if you tried to run the same application on a machine which had the higher version of the .Net framework, say 3.5. The application would still work.
This is because with every release, Microsoft ensures that older framework versions gel well with the latest version.

2. Portability- Applications built on the .Net framework can be made to work on any Windows platform. And now in recent times, Microsoft is also envisioning to make Microsoft products work on other platforms, such as iOS and Linux.
3. Security - The .NET Framework has a good security mechanism. The inbuilt security mechanism helps in both validation and verification of applications. Every application can explicitly define their security mechanism. Each security mechanism is used to grant the user access to the code or to the running program.
4. Memory management - The Common Language runtime does all the work or memory management. The .Net framework has all the capability to see those resources, which are not used by a running program. It would then release those resources accordingly. This is done via a program called the "Garbage Collector" which runs as part of the .Net framework.

   The garbage collector runs at regular intervals and keeps on checking which system resources are not utilized, and frees them accordingly.

5. Simplified deployment - The .Net framework also have tools, which can be used to package applications built on the .Net framework. These packages can then be distributed to client machines. The packages would then automatically install the application.

Summary

- .Net is a programming language developed by Microsoft. It was designed to build applications which could run on the Windows platform.
- The .Net programming language can be used to develop Forms based applications, Web based applications, and Web services. Developers
- can choose from a variety of programming languages available on the .Net platform. The most common ones are VB.Net and C#.

Chapter 2: C# and .Net Version History

.Net Framework Version History

The first version of the .Net framework was released in the year 2002. The version was called .Net framework 1.0. The .Net framework has come a long way since then, and the current version is 4.7.1.

Below is the table of .Net framework versions, which have been released with their release dates. Every version has relevant changes to the framework.

For example, in framework 3.5 and onwards a key framework called the **Entity framework** was released. This framework is used to change the approach in which the applications are developed while working with databases.

Version number	**CLR version**	**Release date**
1.0	1.0	2002-02-13
1.1	1.1	2003-04-24
2.0	2.0	2005-11-07
3.0	2.0	2006-11-06
3.5	2.0	2007-11-19
4.0	4	2010-04-12
4.5	4	2012-08-15
4.5.1	4	2013-10-17
4.5.2	4	2014-05-05
4.6	4	2015-07-20

4.6.1	4	2015-11-17
4.6.2	4	2016-08-02
4.7	4	2017-04-05
4.7.1	4	2017-10-17

The biggest advantage of the .Net framework is that it supports Windows platform. Almost everyone works with Windows machines.

Microsoft always ensures that .Net frameworks are in compliance with all the supported Windows operating systems.

C# Version History

Version	**.NET Framework**	**Visual Studio**	**Important Features**
C# 1.0	.NET Framework 1.0/1.1	Visual Studio .NET 2002	First release of C#
C# 2.0	.NET Framework 2.0	Visual Studio 2005	• Generics Partial • types • Anonymous methods • Nullable types Iterators • Covariance and contravariance •
C# 3.0	.NET Framework 3.0\3.5	Visual Studio 2008	• Auto-implemented properties Anonymous • types Query • expressions Lambda • expression Expression • trees Extension • methods
			• Dynamic binding • Named/optional arguments

C# 4.0	.NET Framework 4.0	Visual Studio 2010	• Generic covariant and contravariant Embedded • interop types
C# 5.0	.NET Framework 4.5	Visual Studio 2012/2013	• Asynchronous members Caller • info attributes
C# 6.0	.NET Framework 4.6	Visual Studio 2013/2015	• Static imports • Exception filters • Property initializers • Expression bodied members • Null propagator String • interpolation nameof • operator Dictionary • initializer
C# 7.0	.NET Core	Visual Studio 2017	• Improved performance and productivity • Azure Support AI • Support • Game development • Cross platform • Mobile App Development • Window App Development

Buy Now $9.99