

Introduction to C#

.NET Ecosystem

C#14 & .NET 10



Mohamed Elshafei

What is C# ?

Definition

C# is a modern, object-oriented programming language developed by Microsoft.

Key Facts

 Developed in 2002

 Runs on .NET Platform

 Strongly Typed Language

 Automatic Memory Management

Application Types



Web APPs

ASP Core MVC ,Razor pages



Desktop Apps

Windows Forms, WPF



Mobile Apps

MAUI, Xamarin



Cloud Services

Azure, AWS



Game Development

Unity engine



Web APIs

Building RESTful services

Key Advantages of C#



Strongly Typed Language

Compile-time type checking prevents runtime errors



Automatic Memory Management

Garbage collector handles memory allocation



High Performance

RyuJIT and NativeAOT for optimal performance



Cross-Platform

.NET 5-10 runs on Windows, Linux, and macOS



Built-in async/await

Simplified asynchronous programming



Large Ecosystem

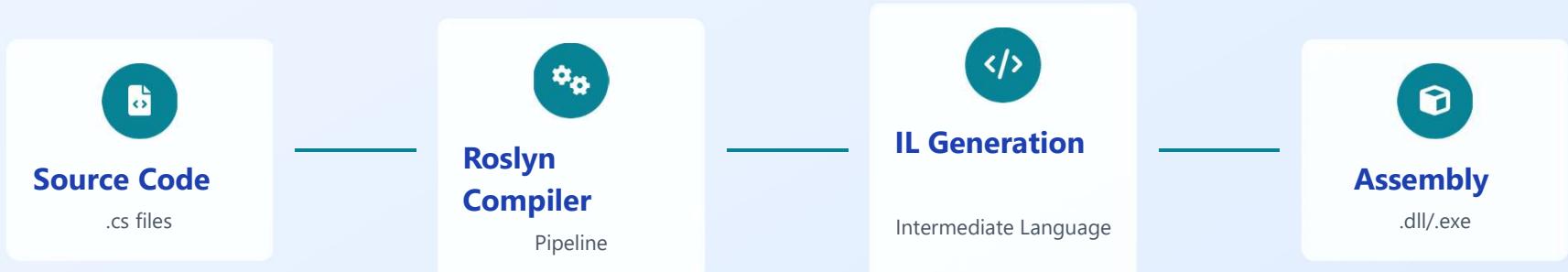
NuGet package manager with millions of packages



Excellent Tools

Visual Studio, Rider, and other IDEs with debugging and profiling

Compilation Process Overview



Source Code

- Developer writes .cs files
- Contains classes and methods

Roslyn Pipeline

- Lexical Analysis
- Syntax Analysis
- Semantic Analysis

IL Generation

- Converts code to IL
- CPU-independent

Assembly Output

- .dll for libraries
- .exe for applications



The compilation process transforms source code into executable code through multiple stages

JIT Compilation

JIT Compilation Process



Types of JIT Compilers

standard/RyuJIT

Default, highly optimized

Econo-JIT

Legacy, less optimization

Pre-JIT/NGen

Compile at installation

Key Benefits of JIT Compilation

- Platform independence through IL
- Optimization at runtime
- Improved security

CLR Execution & JIT Compilation

CLR Components



JIT Compiler

Converts IL to native machine code



Garbage Collector

Manages memory automatically



Exception Manager

Handles errors and exceptions



Security

Enforces code permissions



Interop Services

Enables interaction with unmanaged code (P/Invoke and COM



Class Loader

Loads assemblies, types, and metadata into the runtime



Type System

This enforces strict type checking



Thread Support

This provides multithreading support to applications



Debug Engine

Allow you to debug different type of applications

CLS (Common Language Specification)

common syntax which are supported by all .NET programming languages.

CTS (Common Type System)

Common Data type which is supported by all .NET programming languages

.NET Evolution & Versions



.NET Framework 1.0

Windows only

Windows Forms, WPF, ASP.NET

.NET Core 1-3

Cross-platform

Razor Pages, SPA Templates
Blazor

.NET 5

Unified platform

Single SDK, Performance
Cross-platform

.NET 6

LTS, Minimal APIs

LTS, Minimal APIs, MAUI

.NET 7-9

Performance & Cloud

NativeAOT, Cloud-native, Performance

.NET 10

AI-first runtime

AI Integration, Cloud Optimized, Enhanced AOT

Development Environment Setup

Visual Studio 2026

★ Install Visual Studio



Download Visual studio 2026

Go to <https://visualstudio.microsoft.com/downloads/>
download community version



Required Components

Workloads Individual components Language packs Installation locations

Web & Cloud (4)

- ASP.NET and web development Build web applications using ASP.NET Core, ASP.NET, HTML/JavaScript, and Containers including Docker support.
- Python development Editing, debugging, interactive development and source control for Python.
- Azure development Azure SDKs, tools, and projects for developing cloud apps and creating resources using .NET and .NET Framework...
- Node.js development Build scalable network applications using Node.js, an asynchronous event-driven JavaScript runtime.

Desktop & Mobile (5)

- Mobile development with .NET Build cross-platform applications for iOS, Android or Windows using Xamarin. This includes a preview of the
- .NET desktop development Build WPF, Windows Forms, and console applications using C#, Visual Basic, and F# with .NET and .NET Frame...
- Desktop development with C++ Build modern C++ apps for Windows using tools of your choice including MSVC, Clang, CMake, or MSBuild.
- Universal Windows Platform development Create applications for the Universal Windows Platform with C# XAML or optionally C++.

Code Protection & Obfuscation



Why Obfuscate?

- 🔒 Protect business logic
- 👤 Prevent reverse engineering
- 🔑 Secure sensitive algorithms



Popular Tools

Dotfuscator Community Edition

Free version with basic obfuscation

Babel Obfuscator

Commercial solution with advanced features

ConfuserEx

Open-source .NET obfuscator



What Obfuscation Does



Renames Symbols

Changes class and method names



Encrypts Strings

Protects sensitive text values



Modifies Control Flow

Complicates program logic



Adds Anti-Debugging

Prevents debugging attempts

<https://www.preemptive.com/what-is-obfuscation/>