Introduction to .NET

Introduction to .NET

- .NET is a free, open-source, cross-platform developer framework created by Microsoft.
- It allows developers to build a wide range of applications, including:
  - Desktop apps (WPF, Windows Forms)
  - Web apps (ASP.NET)
  - Mobile apps (.NET MAUI)
  - Cloud services
  - Games (using Unity)
  - IoT apps
  - AI and Machine Learning applications
- There is a variety of programming languages available on the .NET platform, C#, F# and Visual Basic (VB) being the most common ones.
- The first version of .NET Framework 1.0 was released in Feb. 2002.
- The current version of .NET: https://versionsof.net/

Introduction to .NET

- .NET supports multiple programming languages:
  - C# (C-Sharp) – Most Popular:
    - Web apps (ASP.NET Core), desktop apps (WPF, WinForms), mobile apps (.NET MAUI), games (Unity), cloud apps.
  - F# – Functional-first Language:
    - Data science, financial modeling, complex calculations, functional programming.
  - Visual Basic .NET (VB.NET):
    - Primarily used in legacy Windows Forms applications, internal enterprise tools.
  - PowerShell:
    - For scripting on .NET runtime
  - And many more languages…

Two Variants of .NET

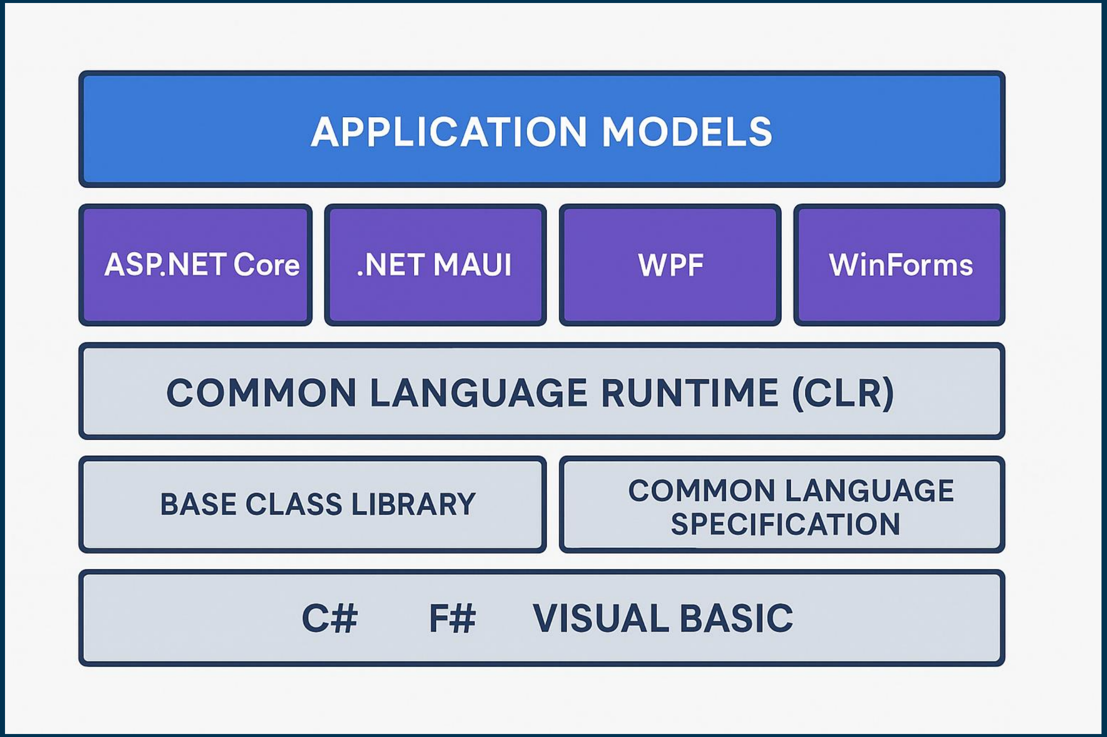
- **.NET Framework**:
  - The original software development platform developed by Microsoft, introduced in the early 2000s.
  - It was designed to build and run Windows-based applications.
  - The last major version is .NET Framework 4.8.1.
  - Microsoft is no longer adding new features, but it's still supported for legacy apps.

- **.NET Core**:
  - It is the cross-platform, open-source version of the original .NET Framework.
  - *.NET Core is now simply called .NET*.
  - The term ".NET Core" applies to versions 1.x to 3.1 (up to 2020).
  - From .NET 5 onwards, everything is unified under one name: **.NET**.

Architecture of .NET

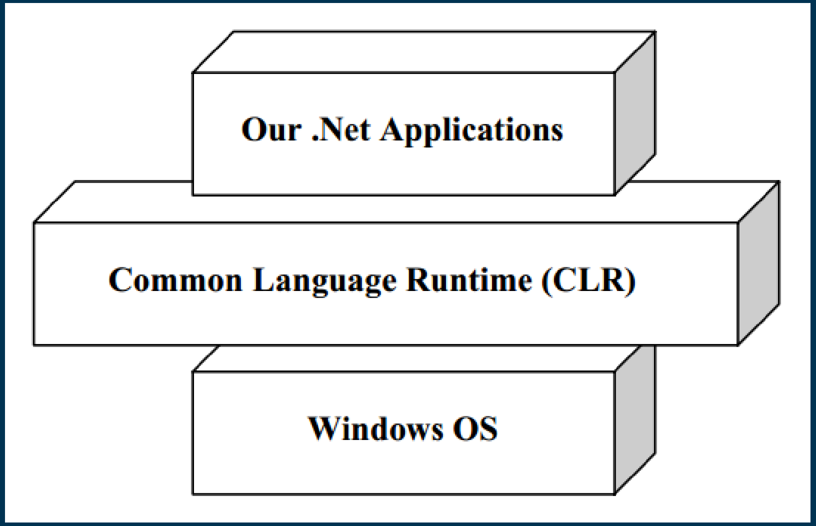
- .NET is a modular and cross-platform framework.
- It’s built in layers, allowing developers to build a wide variety of applications efficiently.

- Let's look at some of these components in the following slides.



Common Language Runtime (CLR)

- The CLR is a virtual machine (VM) – software that manages the execution of programs and hides from them the underlying operating system and hardware.
- It is the *run-time environment* that runs the codes and helps in making the development process easier by providing the various services such as thread management, type-safety, memory management, etc.



- It is responsible for managing the execution of .NET programs regardless of any .NET programming language.
- It resides above the OS and handles the execution of all the .NET applications.
- The programs don't directly communicate with the OS but go through CLR.

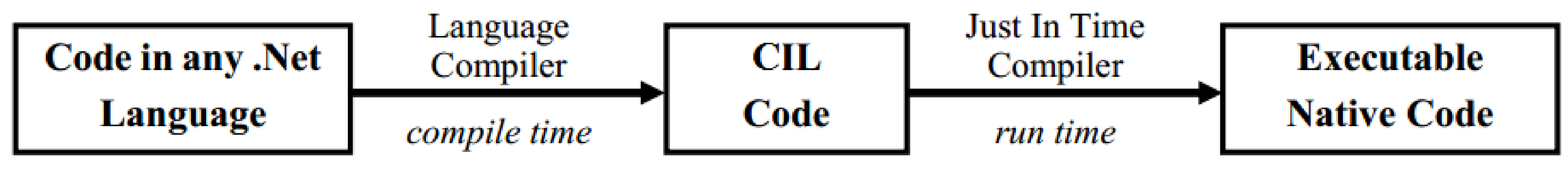
Base Class Library (BCL)

- It is the collection of reusable, object-oriented class libraries, also called the Assemblies.
- It is just like the **header files** in **C/C++** and **packages** in **Java**.
- BCL provides functionality like:
  - File I/O (`System.IO`)
  - Network access (`System.Net`)
  - Collections (`System.Collections`)
  - Database access (`System.Data`)
  - XML handling, LINQ, Regex, etc.

CIL (Common Intermediate Language) Code

(Previously Known as Microsoft Intermediate Language (MSIL))

- When .NET program is compiled using any .NET compliant language (such as C#, VB or C++), the source code does not get converted into the executable binary code, but to an **intermediate code** known as **CIL** which is interpreted by the Common Language Runtime.
- CIL is operating system and hardware independent code.
- Upon program execution, this CIL (intermediate code) is converted to binary executable code (native code).



Code in any .Net Language

Language Compiler

compile time

CIL Code

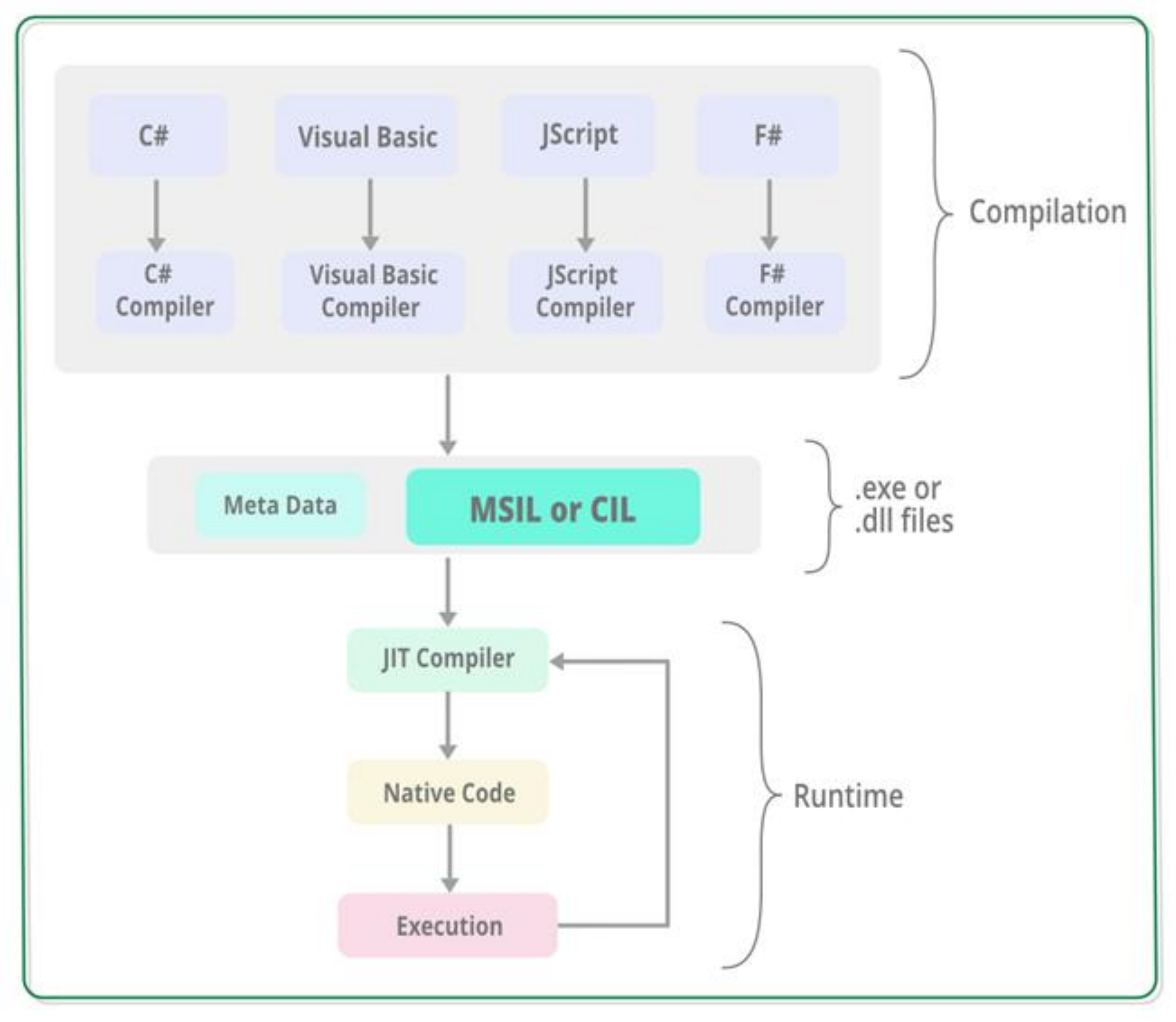
Just In Time Compiler

run time

Executable Native Code

CIL (Common Intermediate Language) Code
(Previously Known as Microsoft Intermediate Language (MSIL))

- The source code is converted into the CIL by a language-specific compiler.
- The CIL is of two types i.e. process assembly (EXE) and library assembly (DLL).
- The JIT compiler then converts the CIL into the machine code.
- It is converted into the machine code on a requirement basis i.e. the JIT compiler compiles the CIL as required rather than the whole of it.
- The machine code obtained using the JIT compiler is then executed by the processor of the computer.



Just-in-Time (JIT) Compiler

- JIT (Just-In-Time) Compiler is a part of the CLR in .NET.
- It converts CIL code (produced when you compile C#, VB.NET, etc.) into native machine code just before execution.
- When the CIL compiled code needs to be executed, the CLR invokes the JIT compiler.
- This native machine code is specific to the computer environment that the JIT compiler runs on.
- JITers are different from traditional compilers as they compile the CIL to native code only when desired.
- For instance, when a method is called, the CIL of the method's body is converted to native code just in time.
  - So, the part of code that is not used by that particular run is never converted to native code.

Common Type System (CTS)

- Common Type System (CTS) describes a set of types (data types) that can be used in different .NET languages.
- The .NET Framework supports cross-language interoperability, which means a language can use the code written in another language.
- For example, a class written in VB.NET can be used in C#.

- CTS ensure that objects written in different .NET languages can interact with each other.
- What this means is that an `int` should mean the same in C#, VB and all other .NET compliant languages.
- For example, CTS defines a type `Int32`, an integral data type of 32 bits (4 bytes) which is mapped by C# through `int` and VB.NET through its `Integer` data type.

Garbage Collection (GC)

- Garbage Collection (GC) is a process managed by the .NET CLR that automatically frees up memory by removing unused or unreferenced objects from the managed heap.
- It prevents memory leaks and improves application performance by managing memory automatically, so developers don’t have to manually delete objects (like in C++).

  - When you create an object using `new`, it's allocated on the managed heap.
  - As your app runs, many objects are created and discarded.
  - Over time, unused objects clutter the memory.
  - That’s when Garbage Collector steps in and cleans up.

WPF (Windows Presentation Foundation)

- Windows Presentation Foundation (WPF) is a graphical subsystem given by Microsoft which uses DirectX and is used in Windows-based applications for rendering UI (User Interface).
- It is a UI framework to create applications with a rich user experience.
- WPF combines application UIs, 2D graphics, 3D graphics, documents and multimedia into one single framework.
- It's vector-based rendering engine uses hardware acceleration of modern graphic cards.
  - This makes the UI faster, scalable and resolution independent.

- Learn More:
  - https://learn.microsoft.com/en-us/dotnet/desktop/wpf/overview/

ASP.NET Core

- ASP.NET Core is a modern, open-source, cross-platform framework for building web applications, APIs, and real-time services.
- Developed by Microsoft, it is a complete rewrite of the older ASP.NET Framework, offering improved performance, modularity, and flexibility.
- ASP.NET Core can run on Windows, Linux, and macOS, making it ideal for cloud and container-based deployments.
- It supports the MVC pattern, Razor Pages, Blazor, and minimal APIs, giving developers multiple options to build scalable web apps.
- Designed to work seamlessly with modern tools and cloud services, it's the go-to choice for web development in the .NET ecosystem.

- Learn More:
  - https://dotnet.microsoft.com/en-us/apps/aspnet

.NET MAUI

- .NET MAUI (Multi-platform App UI) is a modern, cross-platform framework from Microsoft for building native apps for Android, iOS, Windows, and macOS using a single codebase and project structure.
- It allows developers to write UI and business logic in C# and XAML, simplifying cross-platform development.
- .NET MAUI is the evolution of Xamarin.Forms, offering better performance, a unified architecture, and deep integration with the .NET ecosystem.
- With .NET MAUI, you can create responsive, native user interfaces while sharing most of your code across platforms.
- It's ideal for developers building apps for multiple devices with consistent performance and design.

- Learn More:
  - https://dotnet.microsoft.com/en-us/apps/maui

LINQ (Language Integrated Query)

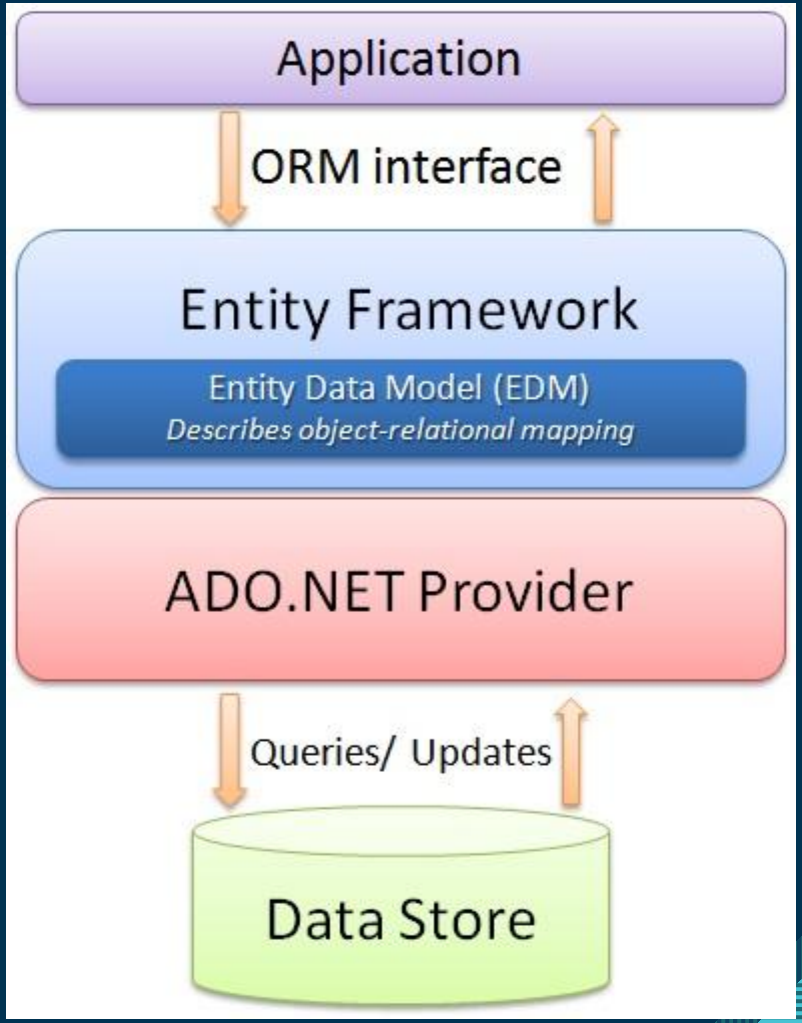
- LINQ (Language Integrated Query) is a feature in .NET that allows you to query data (like collections, databases, XML, etc.) using a SQL like syntax integrated into C#.
- It works with in-memory collections (like arrays, lists), databases (with LINQ to SQL/Entity Framework), XML, and more.
- It makes data manipulation cleaner, readable, and type-safe.
- It eliminates the need for complex loops or manually filtering data.
- It provides compile-time checking and IntelliSense support.

```
int[] numbers = { 1, 2, 3, 4, 5 };

var evenNumbers = from n in numbers
                  where n % 2 == 0
                  select n;

foreach (var num in evenNumbers)
    Console.WriteLine(num);
```

```
Output:

2
4
```

Entity Framework

- It is open-source ORM (Object Relational Mapping) based framework.
- It generates business objects and entities according to the database tables and provides the mechanism for:
  - Performing basic CRUD (Create, Read, Update, Delete) operations.
  - Easily managing "1 to 1", "1 to many", and "many to many" relationships.
  - Ability to have inheritance relationships between entities.
- There are two major versions:
  - EF 6 – for .NET Framework
  - EF Core – lightweight, cross-platform, and works with .NET Core (most recommended)



The C# Language

- C# (pronounced C-Sharp) is a general-purpose, modern and object-oriented programming language.
- It was developed by Microsoft led by Anders Hejlsberg and his team.
- It was approved by the European Computer Manufacturers Association (ECMA) and International Standards Organization (ISO).
- C# is a lot similar to Java syntactically and has roots from the C and C++ languages.

The C# Language

- C# is used for:
  - Mobile applications
  - Desktop applications
  - Web applications
  - Web services
  - Games
  - VR
  - Database applications
  - And much, much more!

Why C#?

- C# has many reasons for being popular and in demand:
  - **Easy to Start:** C# is a high-level language, so it is closer to other popular programming languages like C, C++, and Java and thus becomes easy to learn.
  - **Widely used for developing Desktop and Web Application:** C# is widely used for developing web applications and Desktop applications. If anyone wants to create Microsoft apps, C# is their first choice.
  - **Community:** The larger the community the better it is as new tools and software will be developing to make it better.
  - **Game Development:** C# is widely used in game development and will continue to dominate.

The Visual Studio IDE

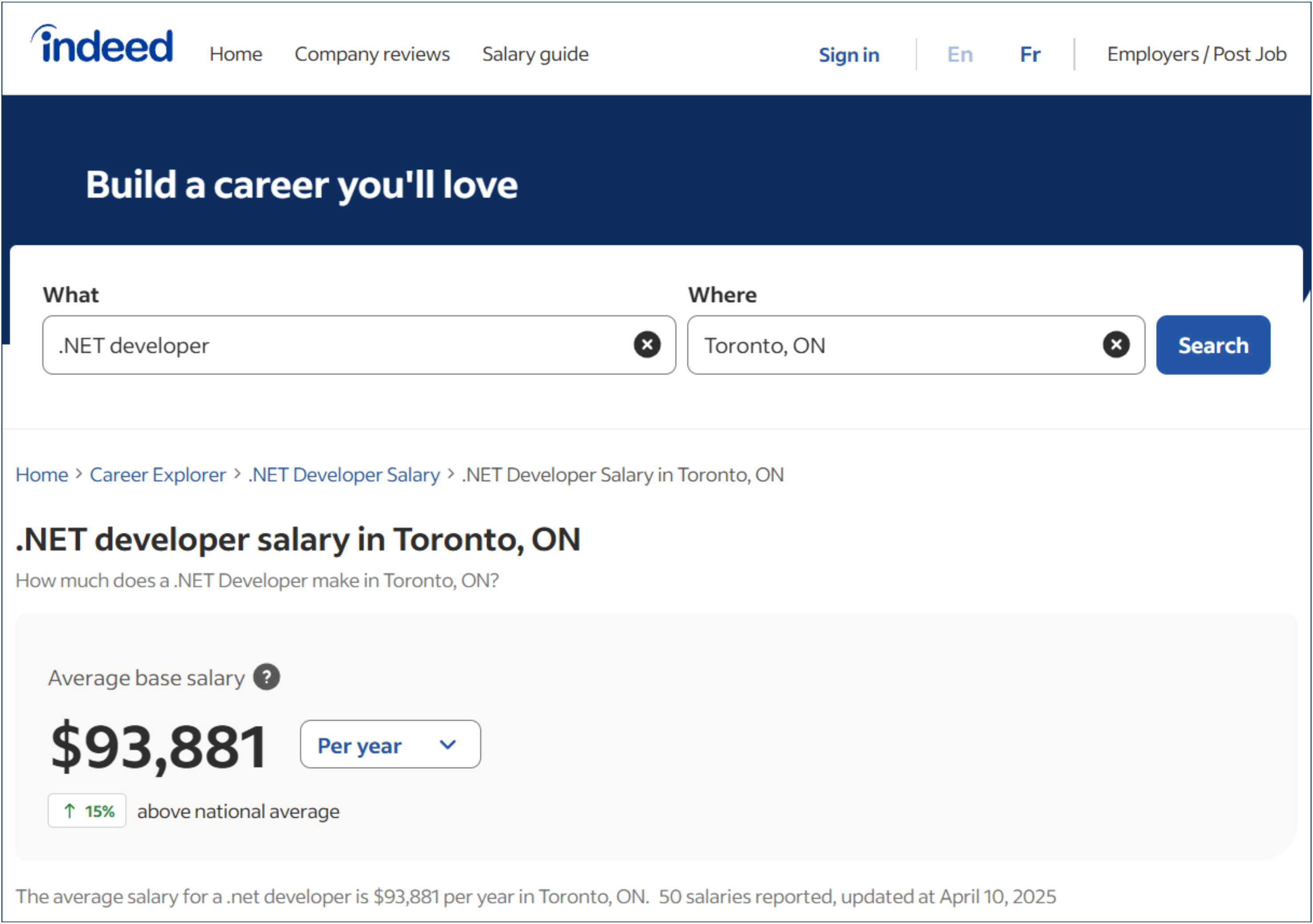
- Visual Studio is an Integrated Development Environment (IDE) released by Microsoft to develop GUI (Graphical User Interface), console, Web applications, mobile apps, cloud, and web services, etc.
- It is not a language-specific IDE so you can use it to write code in C#, C++, VB, Python, JavaScript, and many other languages.
- The Visual Studio Community Edition IDE enables you to write, run, test and debug C# programs quickly and conveniently.

- Its latest version is Visual Studio 2022.
- Version History: https://en.wikipedia.org/wiki/Microsoft_Visual_Studio#History

The Visual Studio IDE

- The IDE provides various useful development tools such as:
  - IntelliSense (auto-complete).
  - Keyword and syntax highlighting
  - Project and solution management.
  - Help building user interface with simple drag and drop support.
  - Properties tab that allow you to set different properties for multiple controls.
  - Hot compiler that checks the syntax of your code as you type it.
  - Compiling and building applications.
  - Standard debugger that allows you to debug your program using putting break points.

Average .NET Developer Salary in GTA



Build a career you'll love

What: .NET developer

Where: Toronto, ON

Home › Career Explorer › .NET Developer Salary › .NET Developer Salary in Toronto, ON

.NET developer salary in Toronto, ON

How much does a .NET Developer make in Toronto, ON?

Average base salary ?

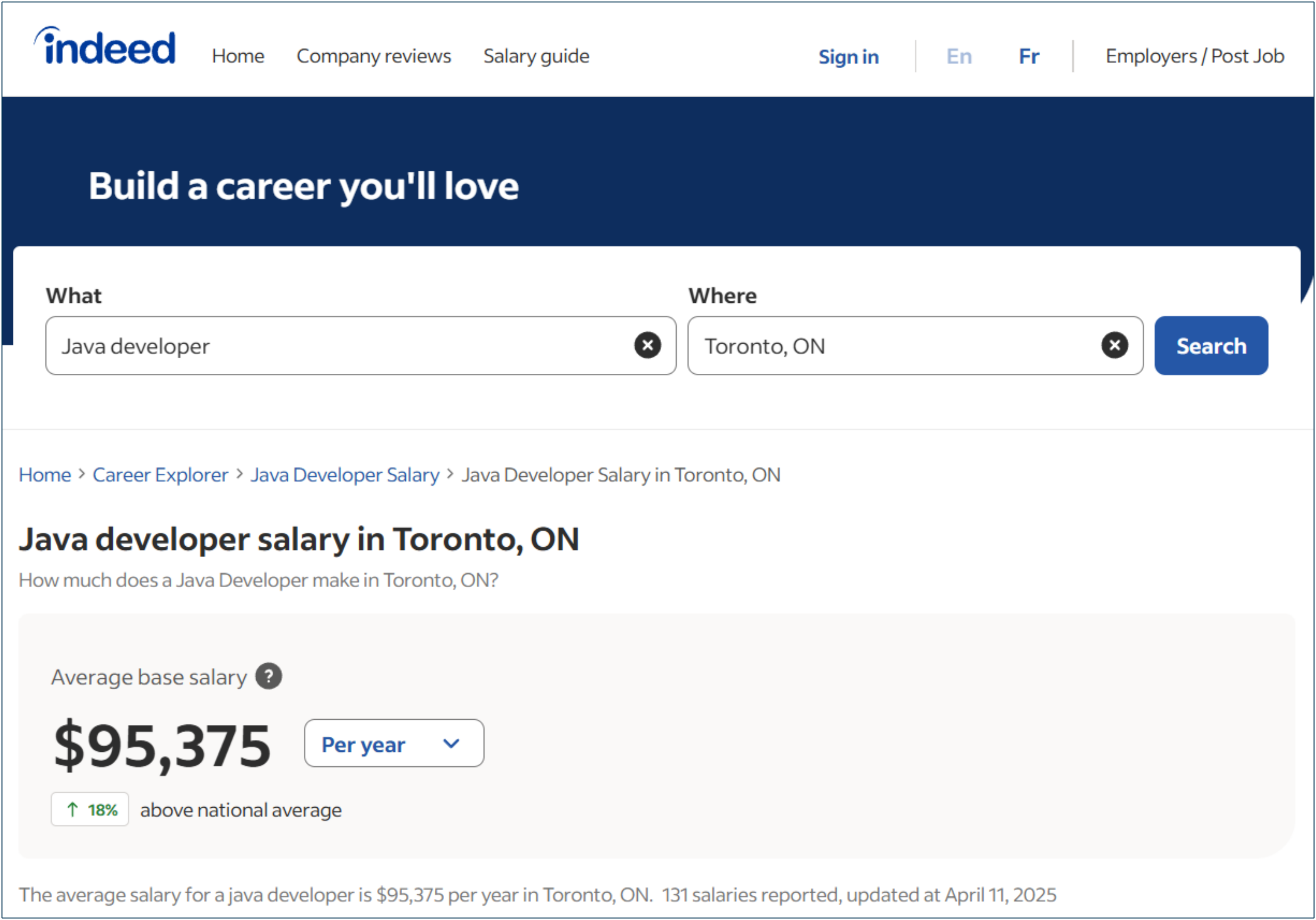
$93,881 Per year

↑ 15% above national average

The average salary for a .net developer is $93,881 per year in Toronto, ON. 50 salaries reported, updated at April 10, 2025

https://ca.indeed.com/career/.net-developer/salaries/Toronto--ON

Average Java Developer Salary in Canada



Build a career you'll love

What

Java developer

Where

Toronto, ON

Home › Career Explorer › Java Developer Salary › Java Developer Salary in Toronto, ON

Java developer salary in Toronto, ON

How much does a Java Developer make in Toronto, ON?

Average base salary ?

$95,375 Per year

↑ 18% above national average

The average salary for a java developer is $95,375 per year in Toronto, ON. 131 salaries reported, updated at April 11, 2025

https://ca.indeed.com/career/java-developer/salaries/Toronto--ON

Thank You

Copyright

The materials provided in class and in SLATE are protected by copyright. They are intended for the personal, educational uses of students in this course and should not be shared externally or on websites such as Chegg, Course Hero or OneClass. Unauthorized distribution may result in copyright infringement and violation of Sheridan policies.