MODULE 1: INTRODUCTION TO VISUAL STUDIO

HISTORY AND EVOLUTION

Visual Studio is one of the most popular tools and used by developers around the world. This integrated development environment (IDE) has evolved significantly since its launch in 1997 and has drastically changed the way developers create applications.

The history of Visual Studio began in **1989**, when Microsoft released its **first integrated development environment** called Visual Basic. This tool quickly became one of the most popular for Windows application development and laid the foundation for what would become Visual Studio.

Current State and Future Directions
- Visual Studio 2022: Focus on performance, reliability, and AI-driven development.
- Continued investment in cloud, mobile, and game development, with integration into Azure and GitHub.

FEATURES AND BENEFITS

1. Multiple Language: Supports various programming languages, including C#, VB.NET, C++, F#, JavaScript, Python, and more.

2. Built in Debugging: Easy troubleshooting of code with breakpoints, watches, and step-through execution.

3. AI Integration: Intelligent code completion, parameter info, quick info, and member lists to speed up coding.

4. User-Friendly: Interface customizable UI that enhances productivity and workflow.

5. Project Template: Re-defined templates for different project types.

6. Git Support: Built-in integration with Git and GitHub for version control.

7. Azure Integration: Integration with Azure services for cloud development and deployment.

8. Cross Platform Support: Create applications for Windows, macOS, and Linux.

INSTALLATION AND SETUP

1. Download: From https://visualstudio.microsoft.com/downloads/ we can download the community or professional and enterprise versions
2. Run Installer: Run the installer file and setup the installation, allow changes.

3. Choose workloads: We can select the necessary workloads that we want to install (e.g., .NET desktop development, ASP.NET, etc.).
4. Install: Click the install button and wait for completion.
5. Launch the visual studio.

SOFTWARE DEVELOPMENT LIFE CYCLE

1. PLANNING: Define project scope, feasibility study, resources needed
2. REQUIREMENTS GATHERING: Collect and document Functional (behaviour and functions of the system, like there should be a login, confirmation mail, CRUD) and Non-Functional (performance, usability, speed, security) requirements.
3. DESIGN: Create user interface and detailed designs
4. IMPLEMENTATION: Coding the system based on the design.
5. TESTING: Perform various tests to identify and fix bugs
6. DEPLOYMENT: Release the software to production, conduct user training, and monitor for issues.
7. MAINTENANCE: Address bugs, provide updates, and ensure ongoing support.

AGILE MODEL

Iterative and Incremental: - Delivering software in small, functional increments, allowing regular feedback adjustments.

1. Planning
2. Req analysis
3. Designing
4. Building
5. Testing

Iterating all these steps

Difference between .net framework and .net core

NET framework helps you build web apps, desktop apps, and web services. It works only on the Windows operating system. On the other hand, .NET core is for creating cross-platform cloud apps that run on Windows, Mac, and Linux.

BASIC STRUCTURE
GETTING STARTED WITH VISUAL STUDIO

Understanding the layout

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
   internal class Program
   {
      static void Main(string[] args)
      {
      }
   }
}
```

What is using?
- In C#, the using directive is used to include namespaces in your code. A namespace is a way to organize code and prevent naming conflicts. By including a namespace, you can access its classes and methods without needing to specify the full path.

What is using System?
- using System; is a directive in C# that includes the System namespace, which is a core part of the .NET framework. This namespace contains essential classes and types, such as Console for input/output operations, String for text manipulation, and Date Time for working with dates and times.
- By including using System; at the beginning of your code, you can access these classes without needing to write the full namespace each time, making your code cleaner and easier to read. It's commonly found in most C# programs since it provides access to many basic functionalities required for development.

- example:
Without using System
System.Console.WriteLine("Hello, World!");

With using System
Console.WriteLine("Hello, World!");