Building a Beginner’s Toolkit for C# (C-Sharp)

---

1. Title & Objective

Technology Chosen

The technology chosen for this toolkit is **C# (C-Sharp)** using the **.NET framework**, developed and executed using **Visual Studio Code**.

Why This Technology

C# is a modern, object-oriented programming language that is beginner-friendly, widely used in industry, and strongly supported by Microsoft. It has clear syntax, strong typing, and excellent documentation, making it suitable for learners who are new to programming.

End Goal

The goal of this project is to create a **beginner-friendly toolkit** that helps new programmers understand fundamental C# concepts through **simple, runnable console applications**. The toolkit aims to reduce the learning curve and provide practical examples that beginners can easily run and modify.

---

2. Quick Summary of the Technology

What is C#?

C# (pronounced *C-Sharp*) is a general-purpose, object-oriented programming language developed by Microsoft. It is commonly used for building desktop applications, web applications, games, and enterprise systems.

Where is it Used?

C# is used in:

- ➔ Desktop applications (Windows applications)

➔ Web development (ASP.NET)

➔ Game development (Unity)

➔ Enterprise and business systems

Real-World Example

A real-world example of C# usage is a **banking system** that manages user accounts, transactions, and security. C# is often used because it is reliable, secure, and scalable.

---

3. System Requirements

To run this toolkit, the following system requirements are needed:

Operating System

- Windows, macOS, or Linux

Tools & Software

- .NET SDK (version 6.0 or higher)
- Visual Studio Code (VS Code)

Editors

- Visual Studio Code with C# extension

Packages

- No external packages required
- Uses built-in .NET libraries only

---

4. Installation & Setup Instructions

Follow the following steps to set up and run the project:

1. Download and install the **.NET SDK** from the official Microsoft website.
2. Install **Visual Studio Code**.
3. Open Visual Studio Code.
4. Open the terminal in VS Code.

5.Create a new console project using the command:

```
dotnet new console -n BeginnersToolkit
```

6.Open the project folder in VS Code.
7.Run the project using:

```
dotnet run
```

---

5. Minimal Working Example

Code Example

```
using System;

class Program
{
    static void Main()
    {
        // Output welcome message
        Console.WriteLine("Welcome to C# beginner toolkit");

        // Declare a variable and assign a value
        int number = 7;

        // Check if the number is greater than 5
        if (number > 5)
        {
            Console.WriteLine("The number is greater than five");
        }
```

```
        else
        {
            Console.WriteLine("Any key to exit");
        }

        // Keeps the console open
        Console.ReadKey();
    }
}
```

Sample Output (when number = 7)

```
Welcome to C# beginner toolkit
The number is greater than five
```

Sample Output (when number = 3)

```
Welcome to C# beginner toolkit
Any key to exit
```

Explanation

1. **`using System;`**
    - This line tells C# to use the `System` library, which contains basic tools like `Console.WriteLine()` for printing text.
2. **`class Program`**
    - Defines a **class** called `Program`. In C#, all code must be inside a class.
3. **`static void Main()`**
    - This is the **entry point** of the program — where execution starts.
    - `static` means you don't need to create an object to run it.
    - `void` means this method doesn't return any value.
4. **`Console.WriteLine("Welcome to C# beginner toolkit");`**
    - Prints the message to the console.

5. `int number = 7;`
    - Declares an **integer variable** named `number` and assigns it the value `7`.
6. `if (number > 5)`
    - Checks a condition: “Is `number` greater than 5?”
    - If **true**, it runs the code inside the first `{ }`.
    - If **false**, it runs the code inside the `else` block.
7. `Console.WriteLine("The number is greater than five");`
    - Runs only if the number is greater than 5.
8. `Console.WriteLine("Any key to exit");`
    - Runs only if the number is 5 or less.
9. `Console.ReadKey();`
    - Waits for the user to press a key so the console doesn’t close immediately.

---

6. AI Prompt Journal

Prompt 1

Prompt:
“Explain what a beginner C# toolkit should include.”

Response Summary:
The AI suggested the development environment,basic project structure,the core language basics,error handling and example of practice programs

Evaluation:
The response was useful but only gave me a wide scope of what is needed .

Prompt 2

Prompt:
"Generate a simple C# console code that:

1.Outputs 'Welcome to C# beginner toolkit'

2. A variable number is declared and a value assigned to it .

3. Using an if condition it checks whether the assigned number is greater than 5 and outputs the number is greater than five if not then it outputs any key to exit.

Ensure the code has comments and a sample outputs for when it is greater than and less than give a brief beginner-friendly explanation."

Response Summary:
The AI generated a simple console program with output, conditionals,comments and an explanation of how the program flows.

Evaluation:
The code was clear, easy to understand, and suitable for the toolkit since I used the SCDD prompting technique which was very efficient.

---

7. Common Issues & Fixes

Issue 1: dotnet command not found

Cause: .NET SDK not installed
Fix: Installed the .NET SDK and restarted the terminal.

Issue 2: Project not running

Cause: Running the command outside the project folder
Fix: Navigated into the correct folder before running `dotnet run`.

Issue 3: Syntax errors

Cause: Missing semicolons or braces
Fix: Reviewed C# syntax and corrected errors.

---

8. References

- Microsoft C# Documentation
- .NET Official Documentation

- Visual Studio Code Documentation
- Beginner C# tutorials from blogs and video tutorials