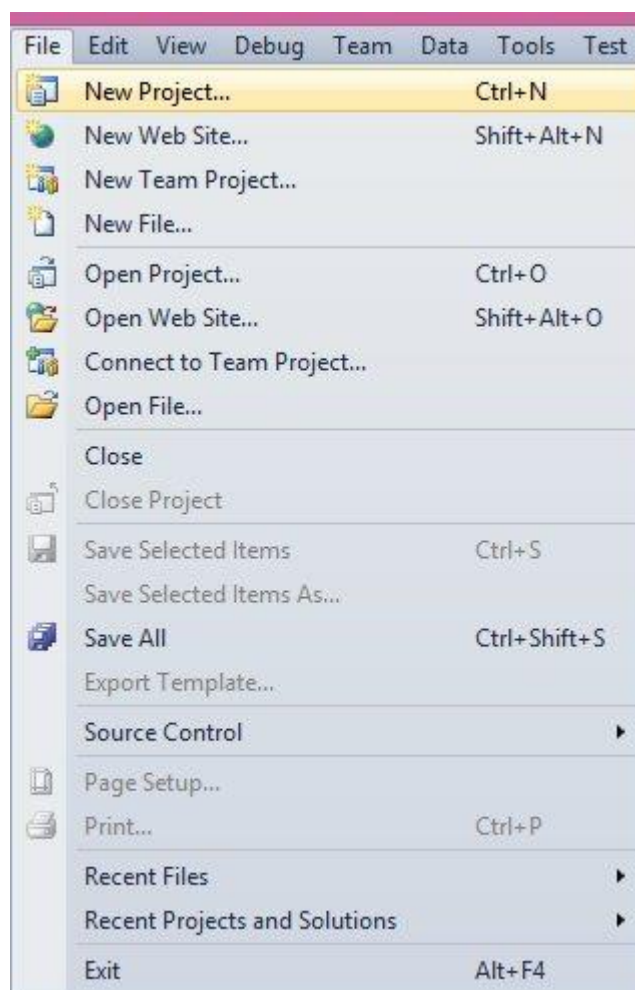


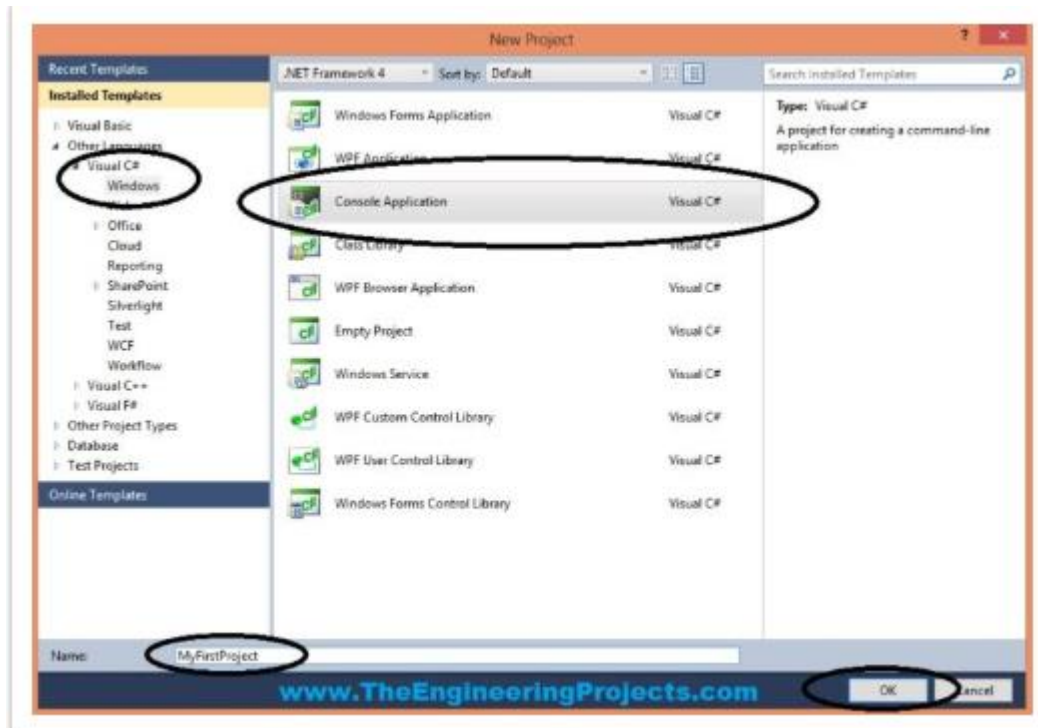
Вступ до C

- C # - мова програмування загального призначення, розроблена Microsoft в 2000 році, як частина .NET Framework. і використовується Microsoft Visual Studio як середовище програмування.
- C # був спочатку розроблений Андерсом Хейльсбергом , який зараз працює головним архітектором C #.
- Я буду використовувати [Microsoft Visual Studio 2019 Community Edition](#) , який є безкоштовним у використанні, і ви можете завантажити його з офіційного веб-сайту.



- У Visual Studio доступні різні мови, тобто C, C ++, C # і F #. C # - найпоширеніша мова програмування серед усіх.
- Тепер я припускаю, що ви встановили Visual Studio і готовий створити ваш перший проект.
- Отже, відкрийте Microsoft Visual Studio і створіть новий проект, натиснувши на **файл** і потім **новий проект**, як показано на малюнку праворуч.

- Ви також можете створити новий проект, натиснувши **Ctrl + N**.
- Після створення нового проекту відкриється нове спливаюче вікно, як показано на малюнку нижче:



- Тепер, як показано на малюнку вище, перш за все виберіть **Visual C#**, тому що будемо використовувати мову C#.
- Далі виберіть **Windows**, а потім **консольну програму**.
- Далі нам потрібно дати ім'я цій програмі консолі, яку я **назвав MyFirstProject** і, нарешті, натиснути кнопку **ОК**.
- Тепер у цьому вікні ми збираємось оновити наш код у C#. Наразі він має простий код Hello World, як показано на малюнку нижче:

```
using System;

namespace TEPPProject
{
    References
    class Program
    {
        References
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
    }
}
```

- Давайте розберемося з цим кодом, рядок за рядком:

Декларація простору імен

- Перший рядок коду - "**using System**", тут Система є вбудованим простором імен C#, і ми оголошуємо його у верхній частині нашого коду.
- Наразі ми використовуємо лише 1 простір імен, але пізніше ми будемо додавати набагато більше, і всі ці декларації простору імен стануть у верхній частині нашого коду.
- Ви можете розглядати простір імен як бібліотеку, в якій є різні класи та методи.
- Отже, коли ми оголошуємо це вгорі, то всі його класи стають доступними для використання у нашому коді, тобто Console - це клас простору імен системи.

Namespaces in C#

```
using System;
using System.Threading;
using SchoolA;
using SchoolB;
```

Простір імен проектів

- Далі ми маємо **простір імен TEPProject**, який є простором імен нашого новоствореного проекту, всі наші класи будуть розміщені всередині цього простору імен.
- Ви можете бачити, що в цьому просторі імен є фігурні дужки {}, у яких є весь решта кодів.

Project Namespace

```
namespace TEPProject
{
}
```

Програмний клас

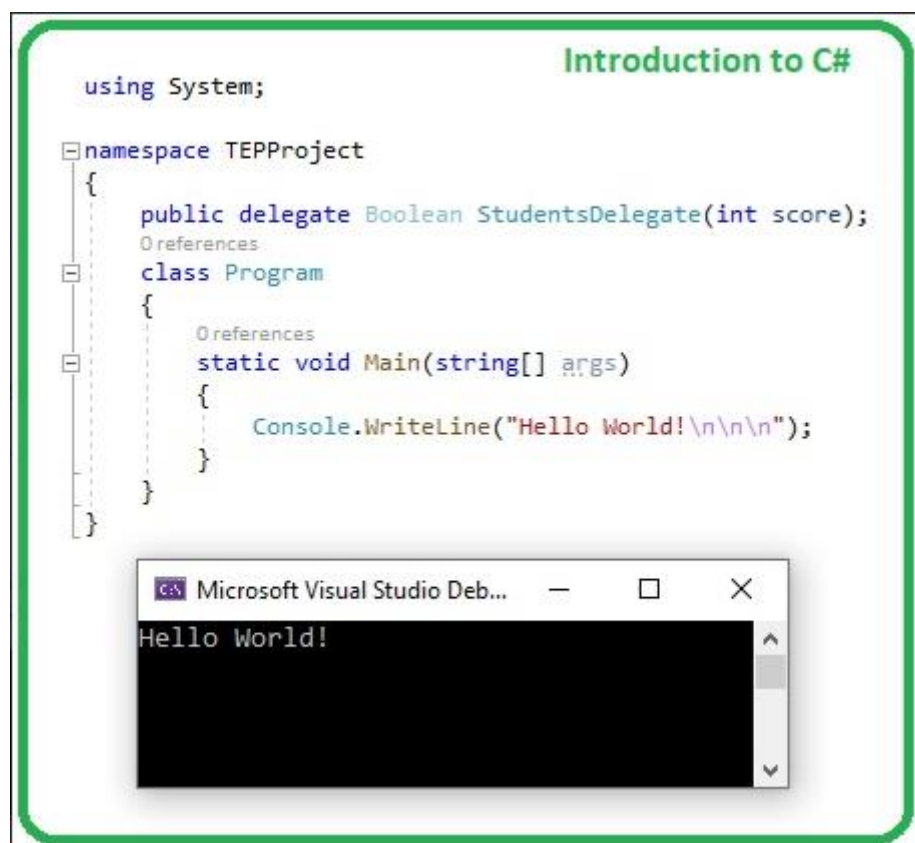
- Коли ви запускаєте свій код, компілятор знаходить простір імен проектів і всередині цього простору імен здійснює пошук класу C# з назвою **Program**, а в клас Program переходить до методу C# з назвою **Main**.

```
class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Hello World!");
    }
}
```

C# Class & Method

- Ось чому у нас є програма **Програма** в нашому просторі імен **TEPProject** і всередині цього класу, у нас є функція **статичного** **недійсного Main (string [] args)**.
- Ця основна функція є статичною функцією і має аргументи з рядком типу даних.
- Ця основна функція є точкою входу нашого компілятора в наш проект, ми повинні написати наш код у цій функції або методі. (Функції також називаються методами)
- Всередині цієї головної функції ми просто надрукували Hello World на нашій консолі.
- Ця консоль є членом Системи імен, якщо ми видалимо простір імен зверху, тоді ця Консоль створить помилку.

Отже, тепер давайте запустимо наш код, і якщо все піде добре, то ви отримаєте щось, як показано на малюнку нижче:

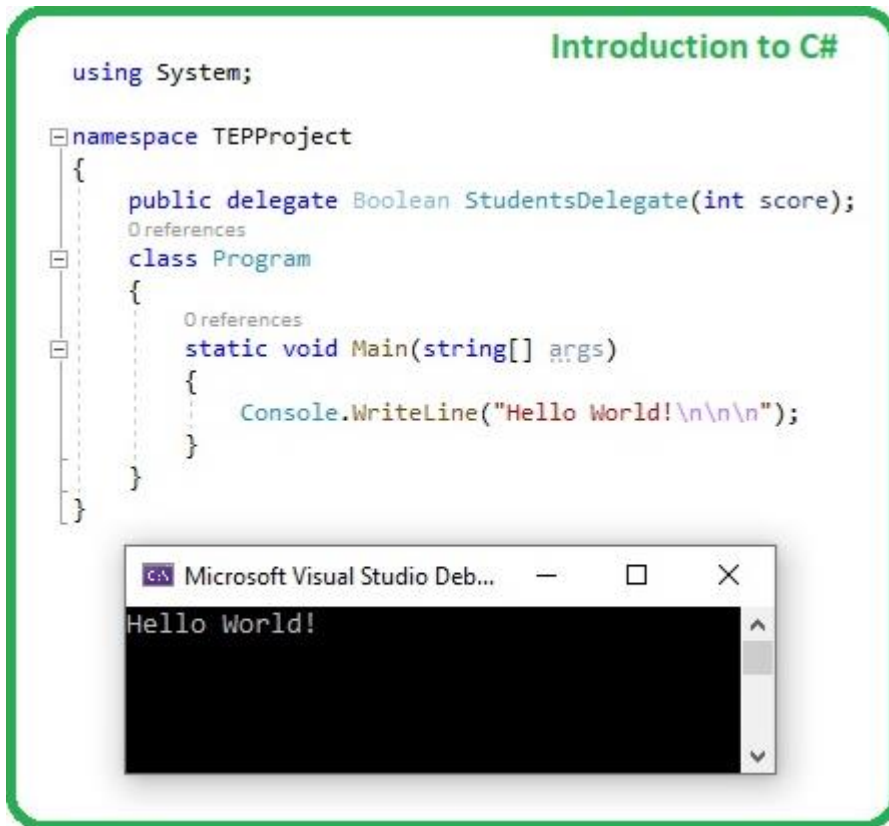


- На нашій консольній панелі друкується Hello World, він більше нічого не робить.

<https://www.theengineeringprojects.com/2016/05/introduction-to-c-sharp.html>

Перший код у C

- На першій лекції ми створили наш перший проект на C #, який друкував **"Hello World!"** На консолі, показаний на малюнку нижче:



- Провідник рішень** містить усі файли вашого проекту, а **Program.cs** - це фактичний файл коду, в який я збираюся написати наш код.
- Отже, додамо наступний код до нашої головної функції файлу Program.cs:

```
Console.WriteLine("What are you learning ???");
string subject = Console.ReadLine();
Console.WriteLine("I am learning {0}", subject);
```
- Тепер ваш файл Program.cs буде виглядати приблизно так, як показано на малюнку нижче:

```
using System;

namespace TEPPProject
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("What are you learning ???");

            string subject = Console.ReadLine();

            Console.WriteLine("I am learning {0}", subject);
        }
    }
}
```

First Code in C#

- У першому рядку коду я друкую на консолі «Що ти вчиш ???».
- У другому рядку я чекаю на вхідні дані з консолі, яку повинен ввести користувач, і я зберігаю ці дані у змінній рядку під назвою subject.
- У третьому рядку я друкую дані, введені користувачем на консолі.
- Ви, мабуть, помітили {0}, це один із способів додавання даних у рядок і називається **синтаксисом власника місця** (Place Holder Syntax).
- Ви можете додати скільки завгодно даних, розділених комами, і можете надрукувати їх як {0} {1} {2} тощо.
- Запустимо наш код і отримаємо аналогічний вихід, як показано на малюнку нижче:

using System;

First Code in C#

namespace TEPPProject

{

public delegate Boolean StudentsDelegate(int score);

References

class Program

{

References

static void Main(string[] args)

{

Console.WriteLine("What are you learning ???");

string subject = Console.ReadLine();

Console.WriteLine("I am learning {0} \n", subject);

}

}

}

Microsoft Visual Studio Deb...

What are you learning ???

C#

I am learning C#

- Я ввів C # і він надрукував його назад, як ви бачите на консолі.
- Тепер, коли ви натиснете будь-яку клавішу, консоль зупиниться.
- Ви також можете використовувати **Concatenation** замість синтаксису власника місць, давайте подивимось на третій рядок, нижче в коді:

using System;

First Code in C#

namespace TEPPProject

{

public delegate Boolean StudentsDelegate(int score);

References

class Program

{

References

static void Main(string[] args)

{

Console.WriteLine("What are you learning ???");

string subject = Console.ReadLine();

Console.WriteLine("I am learning " + subject + "\n");

}

}

}

Microsoft Visual Studio Deb...

What are you learning ???

C#

I am learning C#

- Отже, тепер я використав знак + замість фігурних дужок {}, це називається об'єднанням рядка.
- Якщо запустити свій код, ви отримаєте той самий вихід, як показано на малюнку вище.

Double format specifiers: {0:F2}, {0:N5}, {0:e}, {0:r}, {0:p}, {0:X}, {0:D12}, {0:C}

```
using System;

class MainClass {
    public static void Main() {
        double v = 17688.65849;
        double v2 = 0.15;
        int x = 21;

        Console.WriteLine("{0:F2}", v);

        Console.WriteLine("{0:N5}", v);

        Console.WriteLine("{0:e}", v);

        Console.WriteLine("{0:r}", v);

        Console.WriteLine("{0:p}", v2);

        Console.WriteLine("{0:X}", x);

        Console.WriteLine("{0:D12}", x);

        Console.WriteLine("{0:C}", 189.99);
    }
}
```

```
17688.66
17,688.65849
1.768866e+004
17688.65849
15.00 %
15
000000000021
$189.99
```

Описатель формата	Описание	Примеры	Результат
C или c	Валюта	<code>string s = "\${2.5:C}";</code>	\$2.50
		<code>string s = "\${-2.5:C}";</code>	(\$2.50)
D или d	Decimal	<code>string s = "\${25:D5}";</code>	00025
E или e	Экспоненциальный	<code>string s = "\${250000:E2}";</code>	2.50E+005
F или f	С фиксированной запятой	<code>string s = "\${2.5:F2}";</code>	2.50
		<code>string s = "\${2.5:F0}";</code>	3
G или g	Общее	<code>string s = "\${2.5:G}";</code>	2.5
N или n	Numeric	<code>string s = "\${2500000:N}";</code>	2,500,000.00
P или p	Процент	<code>string s = "\${0.25:P}";</code>	25.00%
R или r	Приемо-передача	<code>string s = "\${2.5:R}";</code>	2.5
X или x	Шестнадцатеричный	<code>string s = "\${250:X}";</code>	FA
		<code>string s = "\${0xffff:X}";</code>	FFFF

Для створення рядка формату використовуйте описувач формату. Рядок формату має вигляд : Axx

- A – це описувач формату, який управляє типом форматування, застосовуванням до числовим значенням;
- xx- показчик точності, який впливає на кількість цифр в форматованому виведення. Значення точності знаходиться в діапазоні від 0 до 99.

Описувачі десяткового ("D" або "d") і шістнадцятирічного форматів ("X" або "x") підтримуються тільки для цілочисельних типів. Описувач формату зворотного перетворення ("R" або "r") підтримується тільки для типів Single , Double і BigInteger .