How to Create a C# Project with Visual Studio Code?

Create a C# Project with Visual Studio Code

In this tutorial, we will learn how to create a C# project and run it in Visual Studio Code.

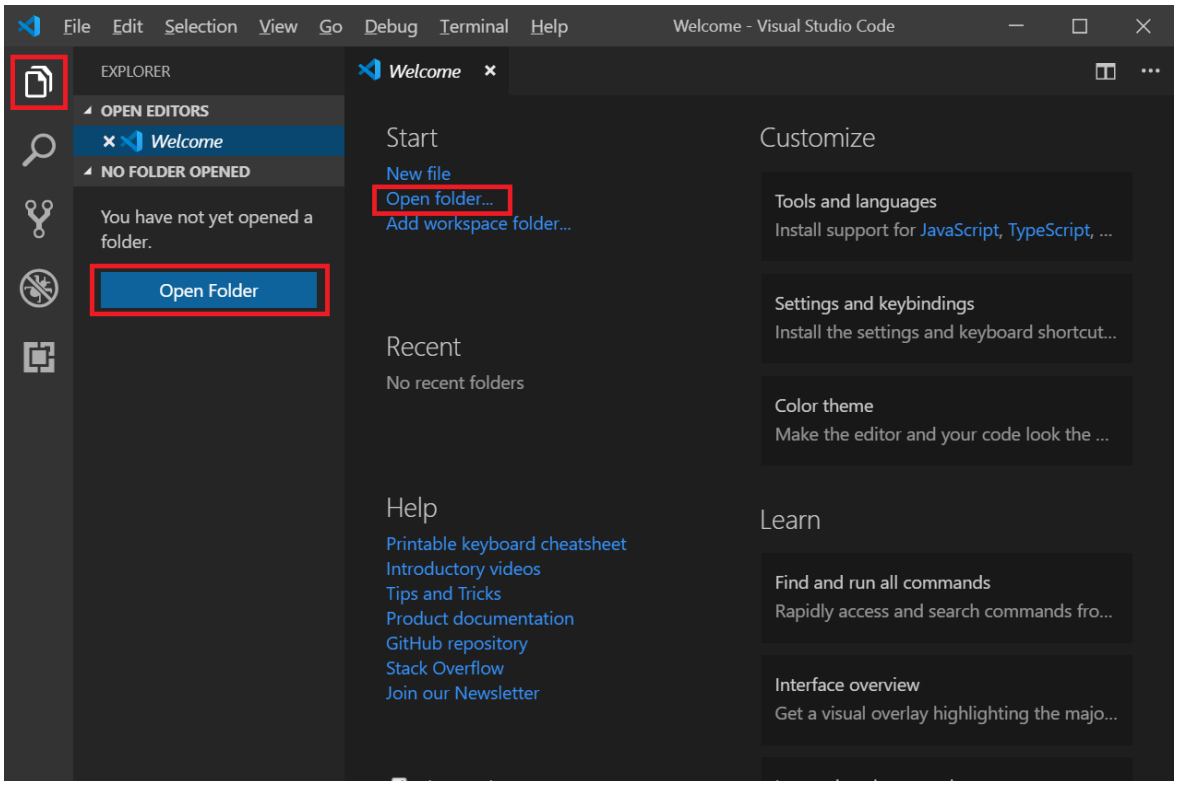
There are three pre-requisites before you can actually get started with this tutorial.

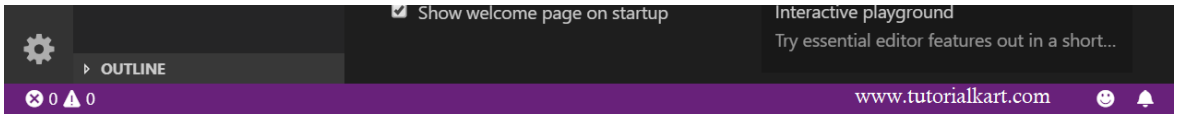
1. Install Visual Studio Code.
2. Install .Net Core.
3. Install C# plugin for Visual Studio Code.

Once all the above requirements are met, we can proceed with the following steps to create a C# project and run it in Visual Studio Code.

1. Open Visual Studio Code

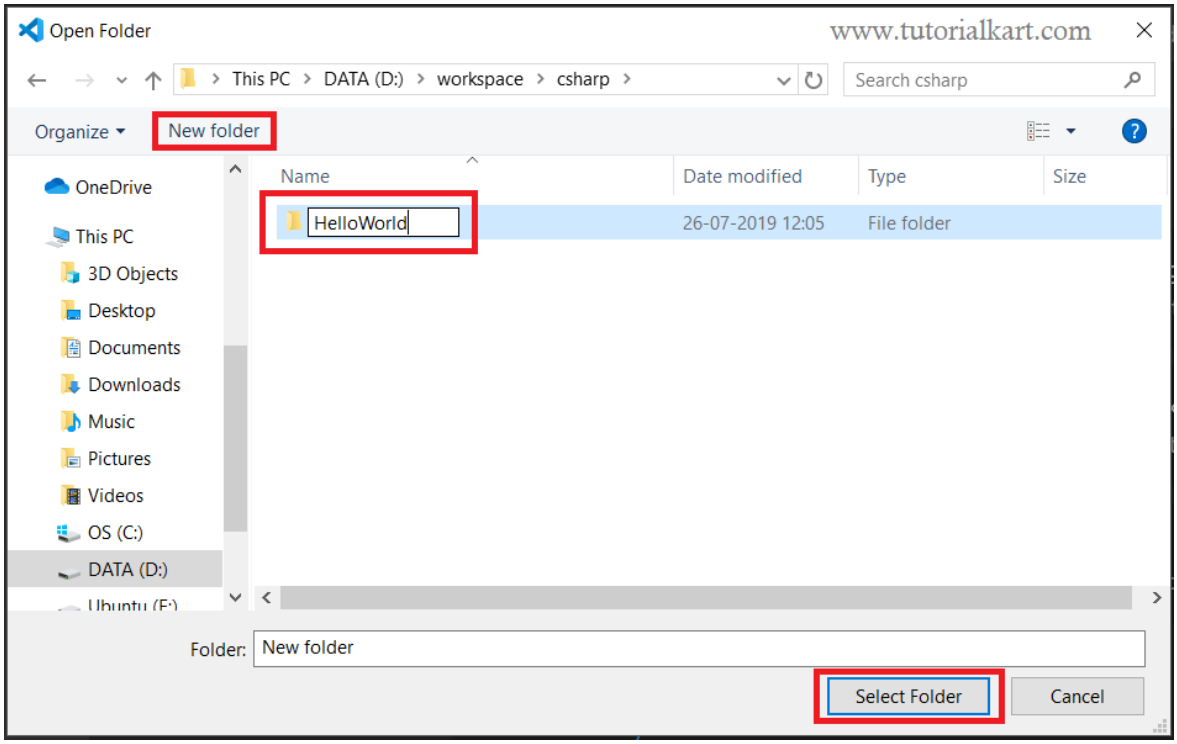
Start Visual Studio Code and you see a **Welcome** page. Click on `Open folder...` link under **Start** section or click on **Explorer** present in the left panel and click on `Open Folder` button.





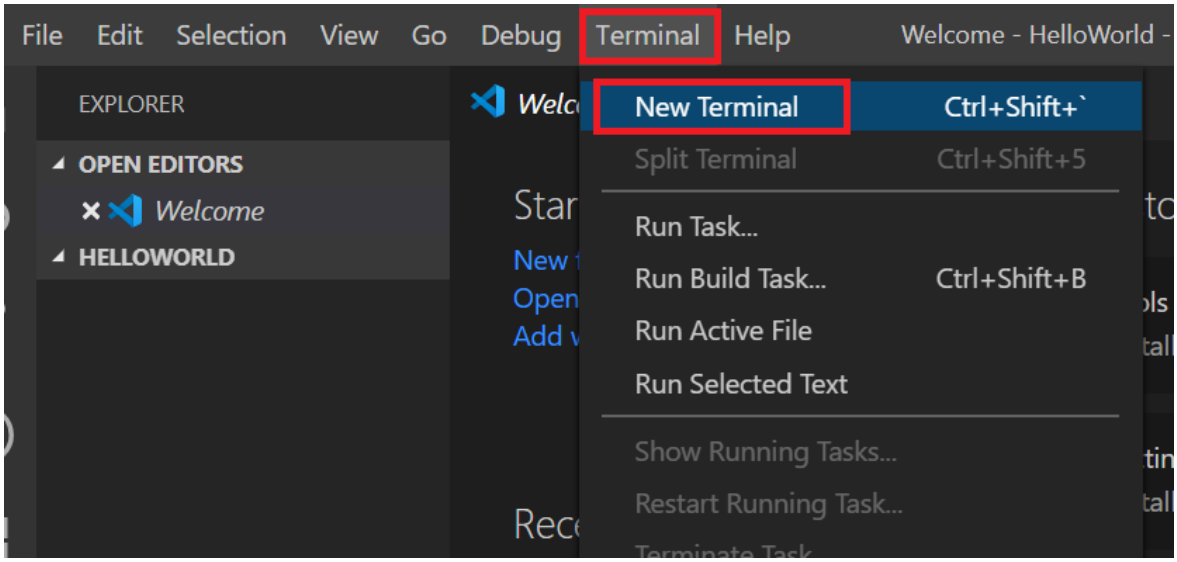
2. Visual Studio Code – Create C# Project – Open Folder

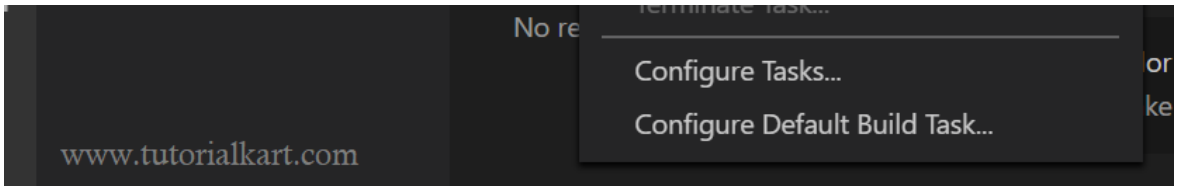
Navigate to the folder in which you would like create project and create a new folder which will be your project. In this tutorial, we will create a C# project named HelloWorld. After you create the folder, click on `Select Folder` button.



3. Visual Studio Code – C# – Terminal

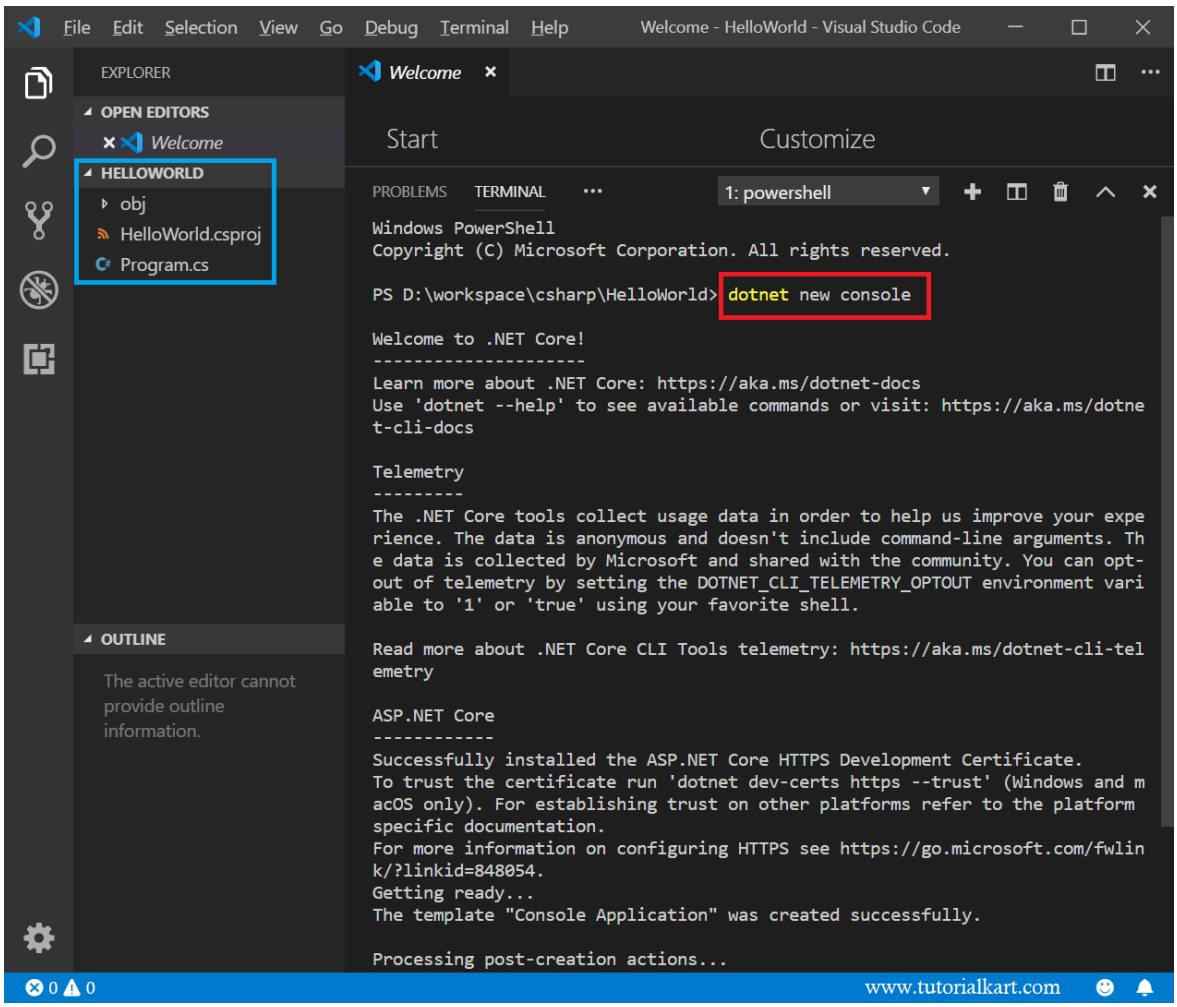
The project is created and the same appears under EXPLORER panel. Now we need to open a new terminal to run some commands to initialize our project and get all the dependencies. Under the **Terminal** menu, click on **New Terminal**.



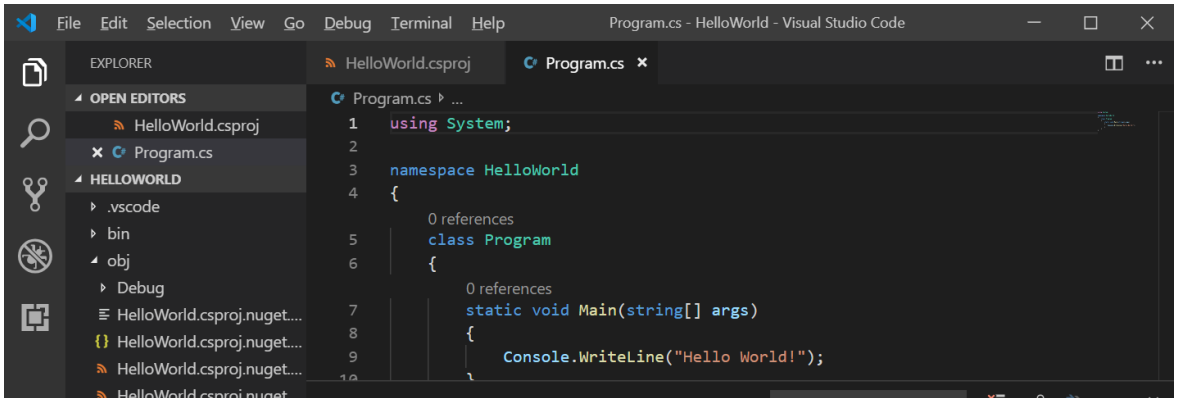


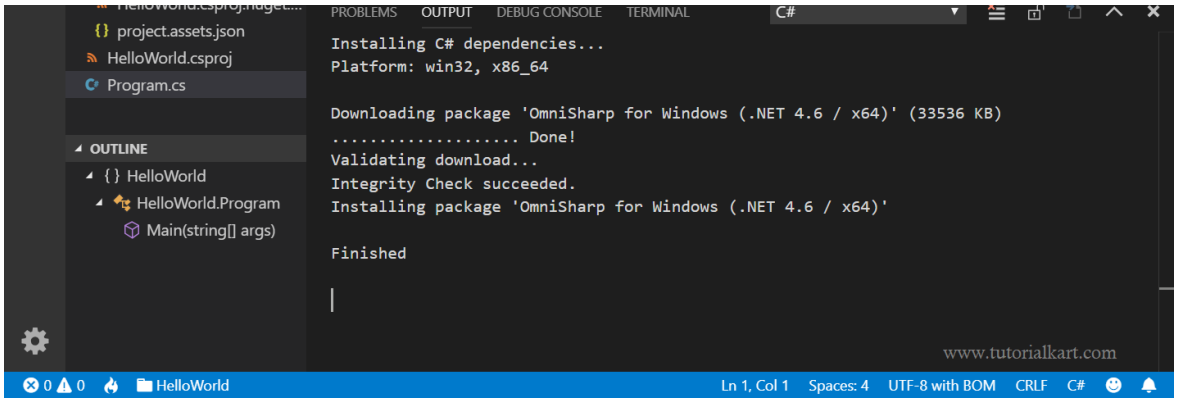
Terminal – dotnet new console

Run the command `dotnet new console` in the terminal. It will create the files **ProjectName.csproj** where ProjectName is the name of the folder we created for this project, **Program.cs**, and the dependencies in **obj** folder.



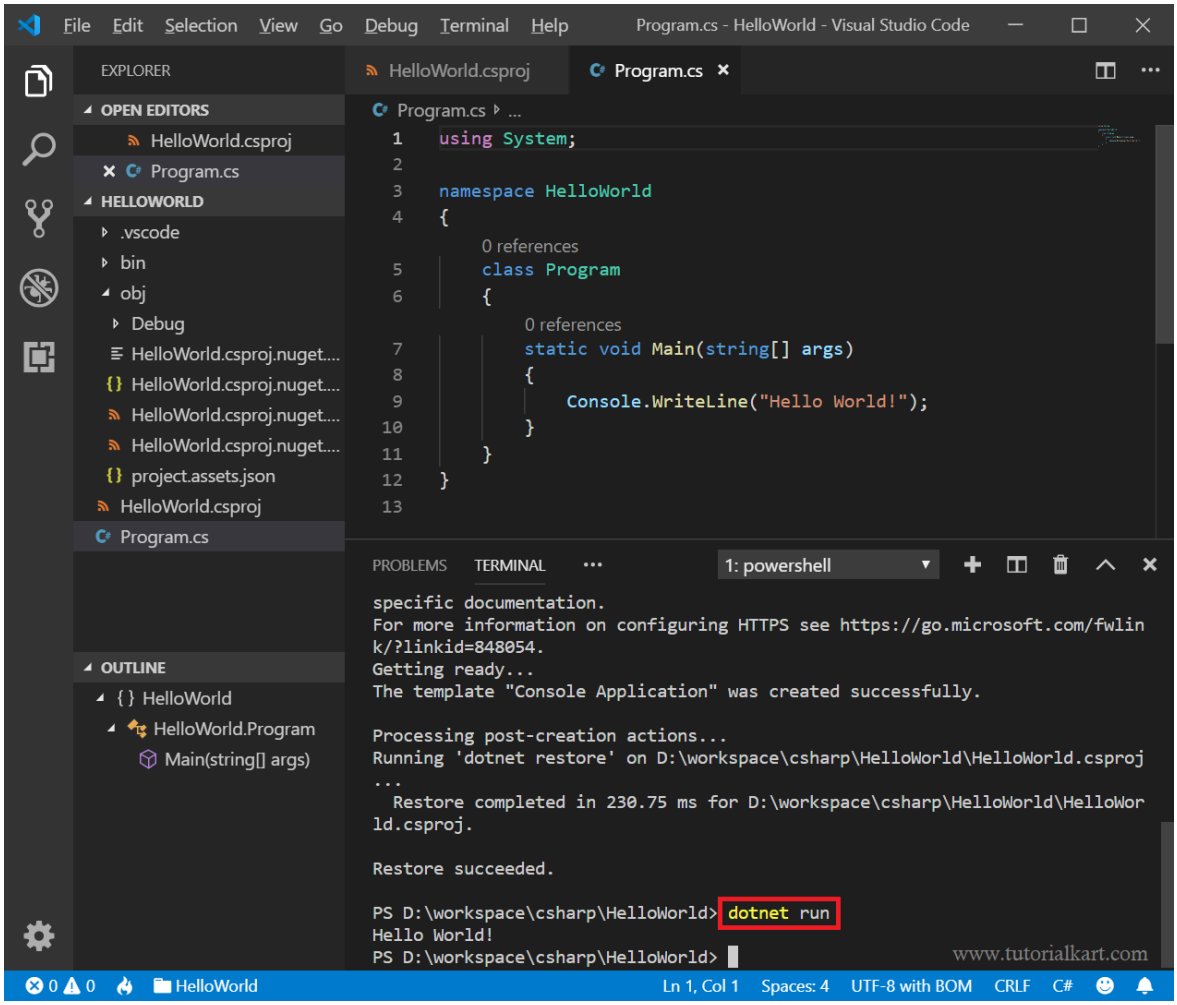
Program.cs contains code to print `Hello World` to print to the console. In the Output section, it is logged that all the required C# dependencies are installed.





4. Terminal – dotnet run

Now, we run the project by running the command `dotnet run` in the terminal.



The project is run and the string is output to the console.

Summary

In this C# Tutorial, We have successfully created the C# project in Visual Studio Code editor.

C# Tutorial

- C# Tutorial
- C# Basic Example
- C# Comments
- C# Variables
- C# Constants
- C# if, if-else
- C# switch
- C# while loop
- C# for loop
- C# foreach
- C# break
- C# continue
- C# struct
- C# enum
- C# String
- C# Array
- C# Command Line Arguments

C# Console Operations

- C# Write to Console
- C# Read from Console

C# Object Oriented Programming Concepts

- C# Class & Constructors
- C# Encapsulation
- C# Polymorphism
- C# Method Overloading
- C# Interfaces

C# String Operations

- C# String Length

- C# Substring

C# File Operations

- C# Read Text File
- C# Write to File
- C# Delete File
- C# Copy File

C# Exception Handling

- C# try-catch
- C# finally
- C# throw
- C# Custom Exception
- C# SystemException
- C# DivideByZeroException
- C# NullReferenceException
- C# InvalidCastException
- C# IOException
- C# FieldAccessException

C# Collections

- C# List
- C# SortedList
- C# HashSet
- C# SortedSet
- C# Stack
- C# Queue
- C# LinkedList
- C# Dicionary
- C# SortedDictionary

C# Errors [Solved]

- C# Error: Class does not contain a constructor that takes arguments