C# Command Line Arguments

C# Command Line Arguments

Command Line Arguments are the arguments that are provided along with the run command in command line or terminal. These arguments can be used to set the initial state of the application.

For example, consider that you built an application, and you have two modes: test, production. In test mode, you print all the debug lines. So, when you start an application, you can provide a command line argument that can make the application run in test or production mode.

Note: Main() function receives the command line arguments as string array `string[]`. You can use this array to access the command line arguments.

Example – Command Line Arguments in C#

In the following example, we have `Main()` function with `string[]` as arguments in its definition. We shall print the number of arguments received and the arguments as well.

Program.cs

```
using System;

namespace CSharpExamples {
    class Program {
        static void Main(string[] args) {
            Console.WriteLine("Number of Arguments : " + args.Length);
            Console.WriteLine("Arguments provided in the command line are: ");
            foreach (string str in args) {
                Console.WriteLine(str);
            }
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run TutorialKart C#
Number of Arguments : 2
Arguments provided in the command line are:
TutorialKart
C#
```

After the command `dotnet run`, we have provided two strings `TutorialKart` and `C#` as arguments. Hence the number of arguments printed is `2`.

Example – Program that does not consider command line arguments

You can define your `Main()` function with no `string[]` as argument. In this case, even if you provide command line arguments, those are not captured as arguments to the `Main()` function.

Program.cs

```
using System;

namespace CSharpExamples {
    class Program {
        static void Main() {
            Console.WriteLine("The program does not consider any arguments");
        }
    }
}
```

Output

```
PS D:\workspace\csharp\HelloWorld> dotnet run
The program does not consider any arguments
PS D:\workspace\csharp\HelloWorld> dotnet run TutorialKart
The program does not consider any arguments
```

Conclusion

In this C# Tutorial, we learned about Command Line Arguments in C#, how to provide these arguments to the program, how to access these values with examples.

- 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