### **2. Join two strings in C#**

We can join two strings in C# using the Concat() method. For example,

// create string1

string str1 = "C# ";

Console.WriteLine("string str1: " + str1);

// create string2

string str2 = "Programming";

Console.WriteLine("string str2: " + str2);

// join two strings

string joinedString = string.Concat(str1, str2);

Console.WriteLine("Joined string: " + joinedString);

****Output****

string str1: C#

string str2: Programming

Joined string: C# Programming

Here, the Concat() method joins str1 and str2 and assigns it to the joinedString variable.

We can also join two strings using the + operator in C#. To learn more, visit C# string Concat.

### **3. C# compare two strings**

In C#, we can make comparisons between two strings using the Equals() method. The Equals() method checks if two strings are equal or not. For example,

// create string

string str1 = "C# Programming";

string str2 = "C# Programming";

string str3 = "Programiz";

// compare str1 and str2

Boolean result1 = str1.Equals(str2);

Console.WriteLine("string str1 and str2 are equal: " + result1);

//compare str1 and str3

Boolean result2 = str1.Equals(str3);

Console.WriteLine("string str1 and str3 are equal: " + result2);

****Output****

string str1 and str2 are equal: True

string str1 and str3 are equal: False

In the above example, we have created 3 strings named str1, str2, and str3. Here, we are using the Equals() method to check if one string is equal to another.

* **Immutability of String Objects:**

In C#, strings are immutable. This means, once we create a string, we cannot change that string.

Consider an example:

// create string

string str = "Hello ";

Now suppose we want to change the string str, add another string "World" to the previous string then

str = string.Concat(str, "World");

Here, we are using the Concat() method to add the string "World" to the previous string str.

**But how are we able to modify the string when they are immutable?**

Let's see what has happened here,

1. C# takes the value of the string "Hello ".
2. Creates a new string by adding "World" to the string "Hello ".
3. Creates a new string object, gives it a value "Hello World", and stores it in str.
4. The original string, "Hello ", that was assigned to str is released for garbage collection because no other variable holds a reference to it.

* **String Escape Sequences:** The escape character is used to escape some of the characters present inside a string. In other words, we use escape sequences to insert special characters inside the string.

Suppose we need to include double quotes inside a string.

string str = "This is the " String " class";

Since strings are represented by double quotes, the compiler will treat "This is the " as the string. And the above code will cause an error.

Now by using **\** before double quote **"**, we can include it in the string.

// use the escape character

string str = "This is the \"String\" class.";

|  |
| --- |
| **Escape Sequence Character Name**  \' single quote  \" double quote  \\ backslash  \0 null  \n new line  \t horizontal tab |

* **String interpolation**

In C#, we can use string interpolation to insert variables inside a string. For string interpolation, the string literal must begin with the $ character.

// create string

string name = "Programiz";

// string interpolation

string message = $"Welcome to {name}";

Console.WriteLine(message);

****Output****

Welcome to Programiz

Notice that,

* the string literal starts with $
* the name variable is placed inside the curly braces {}
* **Methods of C# string**

There are various string methods in C#. Some of them are as follows:

|  |  |
| --- | --- |
| **Methods** | **Description** |
| Format() | returns a formatted string |
| Split() | splits the string into substring |
| Substring() | returns substring of a string |
| Compare() | compares string objects |
| Replace() | replaces the specified old character with the specified new character |
| Contains() | checks whether the string contains a substring |
| Join() | joins the given strings using the specified separator |
| Trim() | removes any leading and trailing whitespaces |
| EndsWith() | checks if the string ends with the given string |
| IndexOf() | returns the position of the specified character in the string |
| Remove() | returns characters from a string |
| ToUpper() | converts the string to uppercase |
| ToLower() | converts the string to lowercase |
| PadLeft() | returns string padded with spaces or with a specified Unicode character on the left |
| PadRight() | returns string padded with spaces or with a specified Unicode character on the right |
| StartsWith() | checks if the string begins with the given string |
| ToCharArray() | converts the string to a char array |
| LastIndexOf() | returns index of the last occurrence of a specified string |