

In VB.Net, you can use strings as array of characters, however, more common practice is to use the String keyword to declare a string variable. The string keyword is an alias for the **System.String** class.

Creating a String Object

You can create string object using one of the following methods:

- By assigning a string literal to a String variable
- By using a String class constructor
- By using the string concatenation operator +
- By retrieving a property or calling a method that returns a string
- By calling a formatting method to convert a value or object to its string representation

The following example demonstrates this:

```
Module strings
  Sub Main()
    Dim fname, lname, fullname, greetings As String
    fname = "Rowan"
    lname = "Atkinson"
    fullname = fname + " " + lname
    Console.WriteLine("Full Name: {0}", fullname)

    'by using string constructor
    Dim letters As Char() = {"H", "e", "l", "l", "o"}
    greetings = New String(letters)
    Console.WriteLine("Greetings: {0}", greetings)

    'methods returning String
    Dim sarray() As String = {"Hello", "From", "Tutorials", "Point"}
    Dim message As String = String.Join(" ", sarray)
    Console.WriteLine("Message: {0}", message)

    'formatting method to convert a value
    Dim waiting As DateTime = New DateTime(2012, 12, 12, 17, 58, 1)
    Dim chat As String = String.Format("Message sent at {0:t} on {0:D}", waiting)
    Console.WriteLine("Message: {0}", chat)
    Console.ReadLine()
  End Sub
End Module
```

When the above code is compiled and executed, it produces the following result:

```
Full Name: Rowan Atkinson
Greetings: Hello
Message: Hello From Tutorials Point
Message: Message sent at 5:58 PM on Wednesday, December 12, 2012
```

Properties of the String Class

The String class has the following two properties:

S.N	Property Name & Description
-----	-----------------------------

Chars

Gets the *Char* object at a specified position in the current *String* object.

2

Length

Gets the number of characters in the current *String* object.

Methods of the String Class

The *String* class has numerous methods that help you in working with the string objects. The following table provides some of the most commonly used methods:

S.N	Method Name & Description
1	Public Shared Function Compare <i>strAAsString, strBAsString As Integer</i> Compares two specified string objects and returns an integer that indicates their relative position in the sort order.
2	Public Shared Function Compare <i>strAAsString, strBAsString, ignoreCaseAsBoolean As Integer</i> Compares two specified string objects and returns an integer that indicates their relative position in the sort order. However, it ignores case if the Boolean parameter is true.
3	Public Shared Function Concat <i>str0AsString, str1AsString As String</i> Concatenates two string objects.
4	Public Shared Function Concat <i>str0AsString, str1AsString, str2AsString As String</i> Concatenates three string objects.
5	Public Shared Function Concat <i>str0AsString, str1AsString, str2AsString, str3AsString As String</i> Concatenates four string objects.
6	Public Function Contains <i>valueAsString As Boolean</i> Returns a value indicating whether the specified string object occurs within this string.
7	Public Shared Function Copy <i>strAsString As String</i> Creates a new <i>String</i> object with the same value as the specified string.
8	Public Sub CopyTo <i>sourceIndexAsInteger, destinationAsChar(, destinationIndex As Integer, count As Integer)</i> Copies a specified number of characters from a specified position of the string object to a

specified position in an array of Unicode characters.

9

Public Function EndsWith *valueAsString* As Boolean

Determines whether the end of the string object matches the specified string.

10

Public Function Equals *valueAsString* As Boolean

Determines whether the current string object and the specified string object have the same value.

11

Public Shared Function Equals *aAsString, bAsString* As Boolean

Determines whether two specified string objects have the same value.

12

Public Shared Function Format *formatAsString, arg0AsObject* As String

Replaces one or more format items in a specified string with the string representation of a specified object.

13

Public Function IndexOf *valueAsChar* As Integer

Returns the zero-based index of the first occurrence of the specified Unicode character in the current string.

14

Public Function IndexOf *valueAsString* As Integer

Returns the zero-based index of the first occurrence of the specified string in this instance.

15

Public Function IndexOf *valueAsChar, startIndexAsInteger* As Integer

Returns the zero-based index of the first occurrence of the specified Unicode character in this string, starting search at the specified character position.

16

Public Function IndexOf *valueAsString, startIndexAsInteger* As Integer

Returns the zero-based index of the first occurrence of the specified string in this instance, starting search at the specified character position.

17

Public Function IndexOfAny *anyOfAsChar()* As Integer

Returns the zero-based index of the first occurrence in this instance of any character in a specified array of Unicode characters.

18

Public Function IndexOfAny *anyOfAsChar(, startIndex As Integer)* As Integer

Returns the zero-based index of the first occurrence in this instance of any character in a specified array of Unicode characters, starting search at the specified character position.

19	Public Function Insert <i>startIndexAsInteger, valueAsString</i> As String Returns a new string in which a specified string is inserted at a specified index position in the current string object.
20	Public Shared Function IsNullOrEmpty <i>valueAsString</i> As Boolean Indicates whether the specified string is null or an Empty string.
21	Public Shared Function Join <i>separatorAsString, ParamArrayvalueAsString()</i> As String Concatenates all the elements of a string array, using the specified separator between each element.
22	Public Shared Function Join <i>separatorAsString, valueAsString(, startIndex As Integer, count As Integer)</i> As String Concatenates the specified elements of a string array, using the specified separator between each element.
23	Public Function LastIndexOf <i>valueAsChar</i> As Integer Returns the zero-based index position of the last occurrence of the specified Unicode character within the current string object.
24	Public Function LastIndexOf <i>valueAsString</i> As Integer Returns the zero-based index position of the last occurrence of a specified string within the current string object.
25	Public Function Remove <i>startIndexAsInteger</i> As String Removes all the characters in the current instance, beginning at a specified position and continuing through the last position, and returns the string.
26	Public Function Remove <i>startIndexAsInteger, countAsInteger</i> As String Removes the specified number of characters in the current string beginning at a specified position and returns the string.
27	Public Function Replace <i>oldCharAsChar, newCharAsChar</i> As String Replaces all occurrences of a specified Unicode character in the current string object with the specified Unicode character and returns the new string.
28	Public Function Replace <i>oldValueAsString, newValueAsString</i> As String Replaces all occurrences of a specified string in the current string object with the specified string and returns the new string.

- 29 **Public Function Split** *ParamArrayseparatorAsChar() As String*
Returns a string array that contains the substrings in the current string object, delimited by elements of a specified Unicode character array.
- 30 **Public Function Split** *separatorAsChar(, count As Integer) As String*
Returns a string array that contains the substrings in the current string object, delimited by elements of a specified Unicode character array. The int parameter specifies the maximum number of substrings to return.
- 31 **Public Function StartsWith** *valueAsString As Boolean*
Determines whether the beginning of this string instance matches the specified string.
- 32 **Public Function ToCharArray As Char**
Returns a Unicode character array with all the characters in the current string object.
- 33 **Public Function ToCharArray** *startIndexAsInteger, lengthAsInteger As Char*
Returns a Unicode character array with all the characters in the current string object, starting from the specified index and up to the specified length.
- 34 **Public Function ToLower As String**
Returns a copy of this string converted to lowercase.
- 35 **Public Function ToUpper As String**
Returns a copy of this string converted to uppercase.
- 36 **Public Function Trim As String**
Removes all leading and trailing white-space characters from the current String object.

The above list of methods is not exhaustive, please visit MSDN library for the complete list of methods and String class constructors.

Examples:

The following example demonstrates some of the methods mentioned above:

Comparing Strings:

```
#include <include.h>
Module strings
Sub Main()
    Dim str1, str2 As String
    str1 = "This is test"
    str2 = "This is text"
    If (String.Compare(str1, str2) = 0) Then
```

```

        Console.WriteLine(str1 + " and " + str2 +
                           " are equal.")
    Else
        Console.WriteLine(str1 + " and " + str2 +
                           " are not equal.")
    End If
    Console.ReadLine()
End Sub
End Module

```

When the above code is compiled and executed, it produces the following result:

```
This is test and This is text are not equal.
```

String Contains String:

```

Module strings
    Sub Main()
        Dim str1 As String
        str1 = "This is test"
        If (str1.Contains("test")) Then
            Console.WriteLine("The sequence 'test' was found.")
        End If
        Console.ReadLine()
    End Sub
End Module

```

When the above code is compiled and executed, it produces the following result:

```
The sequence 'test' was found.
```

Getting a Substring:

```

Module strings
    Sub Main()
        Dim str As String
        str = "Last night I dreamt of San Pedro"
        Console.WriteLine(str)
        Dim substr As String = str.Substring(23)
        Console.WriteLine(substr)
        Console.ReadLine()
    End Sub
End Module

```

When the above code is compiled and executed, it produces the following result:

```
Last night I dreamt of San Pedro
San Pedro.
```

Joining Strings:

```

Module strings
    Sub Main()
        Dim strarray As String() = {"Down the way where the nights are gay",
                                     "And the sun shines daily on the mountain top",
                                     "I took a trip on a sailing ship",
                                     "And when I reached Jamaica",
                                     "I made a stop"}
        Dim str As String = String.Join(vbCrLf, strarray)
        Console.WriteLine(str)
        Console.ReadLine()
    End Sub
End Module

```

When the above code is compiled and executed, it produces the following result:

Down the way where the nights are gay
And the sun shines daily on the mountain top
I took a trip on a sailing ship
And when I reached Jamaica
I made a stop

Processing math: 100%