Рядки у VB.NET (String)

Загальні положення

Клас System.String вміщує методи, які застосовуються для маніпуляцій із рядками. З використанням методів цього класу ми можемо визначити довжину рядка, виконати пошук підрядка, змінити регістр символів у рядку, порівняти два рядка, розділити рядки на підрядки і виконати низку інших дій.

Після того як екземпляр класу String створено, він не може бути змінений — усі методи класу, які змінюють вміст рядка, повертають новий екземпляр цього класу. Клас StringBuilder, який знаходиться у просторі імен System.Text, використовується для створення рядків, вміст яких може бути модифіковано. В Microsoft.NET перший символ рядка має нульовий індекс.

Властивості класу String

```
Imports System
Imports System.String
  Module Module1
   Sub Main()
      Dim Str As String
      Dim I As Integer
      Str = “Економічна кібернетика"
        While I <= Str.Length - 1
          Console.WriteLine(Str.Chars(I))
          I += 1
        End While
    End Sub
  End Module
```

- Клас String містить дві властивості: властивість Chars(Integer), що повертає символ у зазначеній позиції, і властивість Length, що повертає число символів у рядку.
- У прикладі показане використання властивостей, у якому буде виводитись вміст рядка по одному символу.
- Отримали результат:

Пошук підрядка у рядку IndexOf

```
Imports System
Imports System.String
Module Module1
Sub Main()
Dim Str As String
Dim Ch As Char
Dim Chars As Char() = {“н", “е", “т"}
Str = “Економічна кібернетика"
Ch = “к"
Console.WriteLine(Str.IndexOf(“кіб")) 'returns 11
Console.WriteLine(Str.IndexOf(Ch)) 'returns 1
Console.WriteLine(Str.IndexOf(Chars)) 'returns 16
Console.WriteLine(Str.LastIndexOf(Ch))
    'returns20
End Sub
End Module
```

Метод IndexOf(Char) використовується для пошуку першої копії підрядка у даному рядку. Він повертає початкову позицію підрядка, якщо його знайдено, або –1 — у протилежному випадку.

Метод LastIndexOf() знаходить останній екземпляр підрядка у рядку

Отри

```
11
1
16
20
Press any key
```

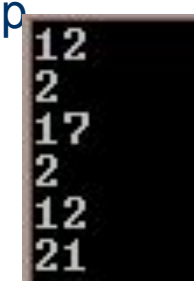
Пошук підрядка у рядку InStr

```
Sub Main()
    Dim Str As String
    Dim Ch As Char
    Dim Chars As Char() = {"н", "е", "т"}
    Dim startPos As Integer
    Str = "Економічна кібернетика"
    Ch = "к"
    startPos = 1
    Console.WriteLine(InStr(startPos, Str, "кіб")) 'returns 12
    Console.WriteLine(InStr(startPos, Str, Ch)) 'returns 2
    Console.WriteLine(InStr(startPos, Str, Chars)) 'returns 17
    Console.WriteLine(InStr(startPos, Str, Ch)) 'returns 2
    startPos = 12
    Console.WriteLine(InStr(startPos, Str, Ch)) 'returns 12
    startPos = 13
    Console.WriteLine(InStr(startPos, Str, Ch)) 'returns 21
End Sub
```

Функція InStr([StartPos,]
String1, string2[, compare])
повертає позицію підрядка string2 у рядку string1. Вона повертає початкову позицію підрядка, якщо його знайдено, або –1 — у протилежному випадку.

Параметр compare задає метод порівняння text або binary. Він дозволяє врахувати регістр символів!!

Проект
FunInStr



Порівняння рядків

Рядки порівнюються для того, щоб визначити чи дорівнюють вони один одному. Якщо ні то один з них "менший" другий – "більший". Визначення більше – менше ґрунтуються на кодах символів ANSII.У випадку літер порядок кодів співпадає з алфавітним порядком – з тим виключенням, що великі літери менше малих (це визначено порядком кодів). Таким чином "ABC" менше "abc" .

Для порівняння двох рядків використовується один із перевантажених версій методу Compare. Цей метод повертає 0, якщо рядки ідентичні, -1, якщо перший рядок менше другого, або +1, якщо перший рядок більше другого. По замовчуванню порівняння рядків виконується з урахуванням регістру символів. Для того щоб регістр символів ігнорувався, третій опціональний параметр методу Compare повинен мати значення True.

Наприклад, три наступних порівняння повертають різні результати:

```
Str = “Microsoft .NET”
Console.WriteLine(Str.Compare(Str.ToUpper, Str.ToLower)) ‘ 1
Console.WriteLine(Str.Compare(Str.ToUpper, Str.ToLower, True)) ‘ 0
Console.WriteLine(str.Compare(str.ToLower, str.ToUpper))  ' -1
```

Перетворення рядка у масив

```
Imports System
Imports System.String
Module Module1
    Sub Main()
        Dim Str As String
        Dim Words() As String
        Dim I As Integer
        Str = "Алгоритмізація та програмування
     процедур обробки інформації"
        Words = Str.Split(" ")
        For I = 0 To Words.GetUpperBound(0)
            Console.WriteLine(I & " : " & Words(I))
        Next
    End Sub
End Module
```

Для перетворення рядка у масив підрядків використовується метод **Split(ParamArray Char()).** Потрібно вказати символ, який вважається символом, що розділяє підрядки, — це може бути пробіл, кома і т.п.

Наступний метод **Join(String, String())** призначено для об'єднання елементів масиву у рядок.

```
0 : Алгоритмізація
1 : та
2 : програмування
3 : процедур
4 : обробки
5 : інформації
Press any key to co
```

Об’єднання масиву у рядок

```
Imports System
Imports System.String
Module Module1
    Sub Main()
        Dim Str, newStr As String
        Dim Words() As String
        Dim I As Integer
        Str = "Алгоритмізація та
      програмування процедур обробки
      інформації"
        Console.WriteLine(Str)
Words = Str.Split(" ")
For I = 0 To Words.GetUpperBound(0)
    Console.WriteLine(I & " : " & Words(I))
Next
Array.Reverse(Words) ‘ reverse масиву Words
newStr = Join(Words, Join(Words, " "))  ‘ об’
      єднали рядок
Console.WriteLine(newStr)
newStr = newStr.ToUpper ‘ верхній регістр
Console.WriteLine(newStr)
newStr = newStr.ToLower ‘ нижній регістр
Console.WriteLine(newStr)
End Sub
End Module
```

```
Алгоритмізація та програмування процедур обробки інформації
0 : Алгоритмізація
1 : та
2 : програмування
3 : процедур
4 : обробки
5 : інформації
інформації обробки процедур програмування та Алгоритмізація
ІНФОРМАЦІЇ ОБРОБКИ ПРОЦЕДУР ПРОГРАМУВАННЯ ТА АЛГОРИТМІЗАЦІЯ
інформації обробки процедур програмування та алгоритмізація
Press any key to continue_
```

Заміна символів/фраз та конкатенація

```
Dim Str, newStr As String
Dim Str1, Str2, Str3 As String
Str = "Спеціальність економічна кібернетика"
Console.WriteLine(Str)
    newStr = Str.Replace(" ", "_") ' замінили пробіл на _
    Console.WriteLine(newStr)
    Str = newStr.Replace("Спеціальність", "Фах") '
     замінили спец на фах
    Console.WriteLine(Str)

    Str1 = "Університет "
    Str2 = "споживчої "
    Str3 = "кооперації"
    Str = Str1.Concat(Str1, Str2, Str3) ' конкатенація
     рядків
    Console.WriteLine(Str)
    newStr = Str1 & Str2 & Str3 ' можна і так
    Console.WriteLine(newStr)
```

Для заміни усіх екземплярів вказаного символа на інший символ використовуємо метод **Replace(Char, Char)** або **Replace(String, String)**

Для об'єднання однієї або декількох рядків у один рядок використовується один із перевантажених методів **Concat**.

```
Спеціальність економічна кібернетика
Спеціальність_економічна_кібернетика
Фах_економічна_кібернетика
Str=    Університет споживчої кооперації
newStr= Університет споживчої кооперації
Press any key to continue_
```

Доповнення та видалення символів

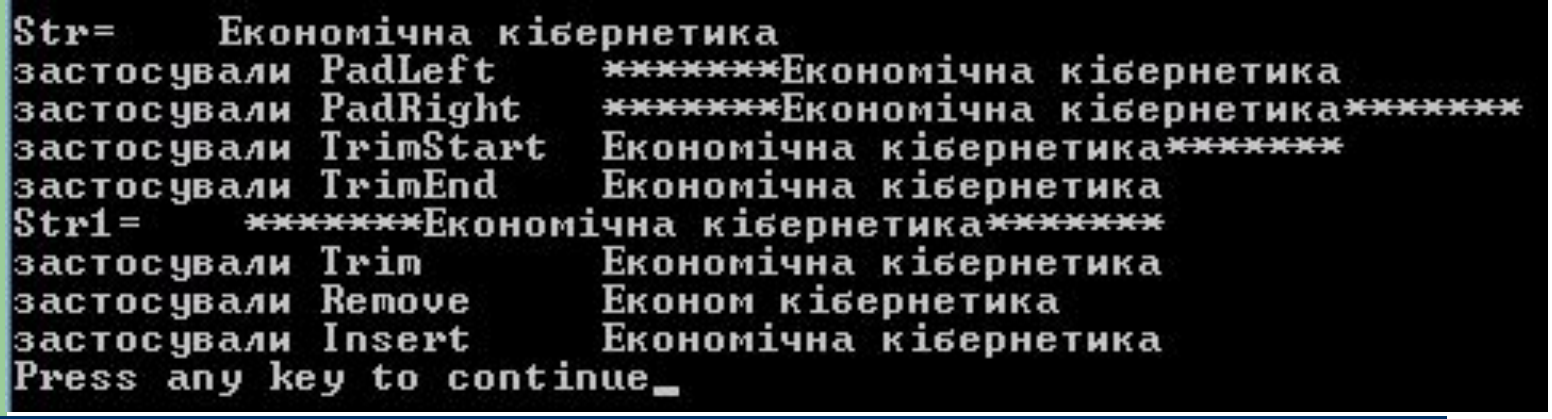
Методи PadLeft(Integer, Char) и PadRight (Integer, Char) використовуються для заповнення рядка ліворуч або праворуч вказаною кількістю символів.

Методи TrimStart(Char()) и TrimEnd(Char()) призначені для видалення заданої послідовності символів з початку або кінця рядка. Метод Trim() використовується для видалення вказаної послідовності символів як із початку, так і з кінця рядка.

Для видалення заданої кількості символів із рядка з вказаної позиції призначено метод Remove(Integer, Integer).

Щоб вставити рядок у вказану позицію іншого рядка, використовується метод Insert(Integer, String).

Приклад застосування вказаних методів – на наступному слайді.




```
Str = "Економічна кібернетика"
    Console.WriteLine("Str=    " & Str)
Str = Str.PadLeft(Str.Length + 7, "*")
    Console.WriteLine("застосували
  PadLeft    " & Str)
Str = Str.PadRight(Str.Length + 7, "*")
    Console.WriteLine("застосували
  PadRight   " & Str)
Str1 = Str
Str = Str.TrimStart("*")
    Console.WriteLine("застосували
  TrimStart  " & Str)
Str = Str.TrimEnd("*")
```

```
Console.WriteLine("застосували
  TrimEnd    " & Str)
Console.WriteLine("Str1=    " & Str1)
Str1 = Str1.Trim("*")
    Console.WriteLine("застосували
  Trim       " & Str1)
Str = Str.Remove(Str.IndexOf("ічна"),
  "ічна".Length)
    Console.WriteLine("застосували
  Remove     " & Str)
Str = Str.Insert(Str.IndexOf("кіб") - 1,
  "ічна")
    Console.WriteLine("застосували
  Insert     " & Str)
```

Вирізання підрядка із рядка

Для вирізання підрядка із рядка застосовується метод Substring(StartIndex, Length) або функція Mid(String, Start, Length)

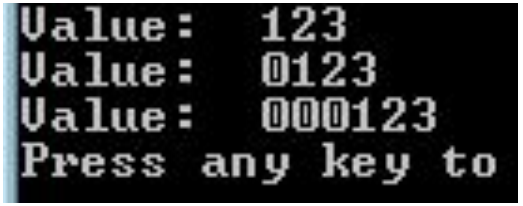
```
Dim Str, Str1 As String
Str = "Економічна кібернетика“
Str1 = Str.Substring(11, 5)
Str1 = Mid(Str, 12, 5)
```

У обох випадках значення Str1 буде дорівнювати “кібер”

Форматування типу Decimal

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {"D", "D4",
    "D6"}
Dim I As Integer
Dim Value As Integer = 123
For I = 0 To
    Formats.GetUpperBound(0)
    Console.WriteLine("Value: {0:" &
    Formats(I) & "}", Value)
Next
End Sub
End Module
```

Це форматування вказується специфікатором D (або d). Також можна вказати точність виведення — мінімальну кількість цифр.



Форматування типу Exponential

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {"E", "E2",
    "E3"}
Dim I As Integer
Dim Value As Integer = 123456
For I = 0 To Formats.GetUpperBound(0)
Console.WriteLine("Value:  {0:" &
    Formats(I) & "}", Value)
Next
End Sub
End Module
```

Використовується формат (який також називається «інженерним» форматом; специфікатор E або e) для перетворення значень у експоненціальне представлення. Також можемо вказати точність перетворення — кількість символів після десяткової точки

```
Value:  1.234560E+005
Value:  1.23E+005
Value:  1.235E+005
```

Форматування типу Fixed-point

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {"F", "F3",
    "F5"}
Dim I As Integer
Dim Value As Integer = 123
For I = 0 To
    Formats.GetUpperBound(0)
Console.WriteLine("Value:  {0:" &
    Formats(I) & "}", Value)
Next
End Sub
```

Форматування чисел із фіксованою точкою (специфікатор F або f) використовується для перетворення десяткових чисел шляхом додавання вказаної кількості нулів (по замовчуванню два) після десяткової точки.

```
Value:  123.00
Value:  123.000
Value:  123.00000
```

Форматування типу Number

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {"N", "N3",
    "N5"}
Dim I As Integer
Dim Value As Integer = 123456
For I = 0 To Formats.GetUpperBound(0)
Console.WriteLine("Value:  {0:" &
    Formats(I) & "}", Value)
Next
End Sub
End Module
```

Цей формат (специфікатор N або n) використовується для перетворення значень у форму [-]d,ddd,ddd.dd.

```
Value:  123,456.00
Value:  123,456.000
Value:  123,456.00000
```

Форматування типу Percent

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {"P", "P3", "P5"}
Dim I As Integer
Dim Value As Decimal = 0.12345
For I = 0 To Formats.GetUpperBound(0)
Console.WriteLine("Value:  {0:" &
    Formats(I) & "}", Value)
Next
End Sub
End Module
```

Використовують цей формат (специфікатор P або p) для перетворення числового значення у вигляді процента.

```
Value:   12.35 %
Value:   12.345 %
Value:   12.34500 %
```

Форматування типу Picture Numeric

```
Imports System
Module Module1
Sub Main()
Dim Formats() As String = {" {0:0.##}", "
    {0:##.###}",  "{0:%#.##}", "{0:##.##E+0}",
    "{0:{{##.##}}}",   "{0:\###.##\#}"}
Dim I As Integer
Dim Value As Double = 1.23456
For I = 0 To Formats.GetUpperBound(0)
Console.WriteLine("Value:  " & Formats(I),
    Value)
Next
End Sub
End Module
```

На попередніх слайдах використовувалися шаблони з різними специфікаторами форматів. Окрім простих шаблонів можна використовувати і більш комплексні. Деякі з таких шаблонів продемонстровано у прикладі:

```
Value:    1.23
Value:    1.235
Value:   %123.46
Value:   12.35E-1
Value:   (1.23)
Value:   #1.23#
```