(T10)比較 SystemString、SystemTextStringBuilder。Ram 的 Stack、Heap
CourseGUID: 29f1196a-1950-41a4-b9c1-dd13a9e92d92
==========
(T10)比較 SystemString、SystemTextStringBuilder。Ram 的 Stack、Heap
==========

==========

0. Summary

1.
System.String V.S. System.Text.StringBuilder

1.1.
System.Text.StringBuilder object is mutable
and good for performance,
but System.String is immutable.
Therefore, heavy string manipulation should
use System.Text.StringBuilder.

1.2.
System.String
```
//System.String
//string str1 = "IT";
//str1 += "Handy";
//str1 += "Guy";
//str1 += " Tutorial";
//str1 += " is";
//str1 += " awesome.";
//Console.WriteLine(str1);
```
1.2.1.
RAM contains Stack and Heap area.
Stack is for storing object reference variable.
Heap is for storing the object.
* Firstly, str1 pointed "IT"
* Secondly, str1 pointed "ITHandy",
and "IT" became an orphaned object in heap until it is garbage collected.

* Thirdly, str1 pointed "ITHandyGuy",
and "ITHandy" became an orphaned object in heap until it is garbage collected.
* Do so until str1 finally is pointing "ITHandyGuy Tutorial is Awesome.",
and rest of string objects became orphaned objects in heap until they are garbage collected.

```
   Stack    |   Heap
-----------------------------------------------
            |   "IT"
            |   "ITHandy"
            |   "ITHandyGuy"
            |   "ITHandyGuy Tutorial"
            |   "ITHandyGuy Tutorial is"
 str1   -------> "ITHandyGuy Tutorial is Awesome."
-----------------------------------------
```

1.3.
System.Text.StringBuilder

```
//StringBuilder strBuilder =
//    new StringBuilder("IT");
//strBuilder.Append("Handy");
//strBuilder.Append("Guy");
//strBuilder.Append(" Tutorial");
//strBuilder.Append(" is");
//strBuilder.Append(" awesome.");
//Console.WriteLine(strBuilder.ToString());
```

1.3.1.
No matter how many times the string is manipulated,
StringBuilder always points to the same object instance.

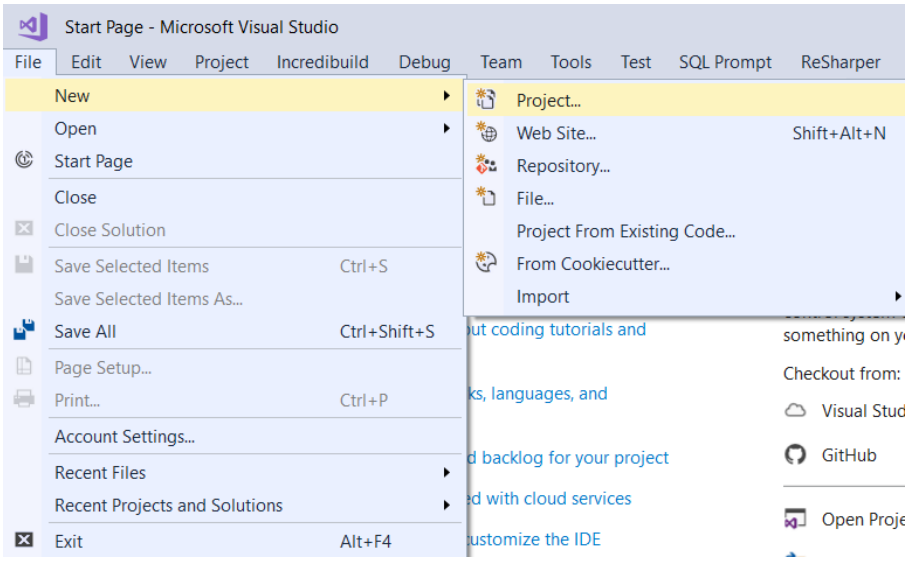
```
   Stack        |   Heap
-------------------------------------------------
                |
 StringBuilder ---> "ITHandyGuy Tutorial is Awesome."
```

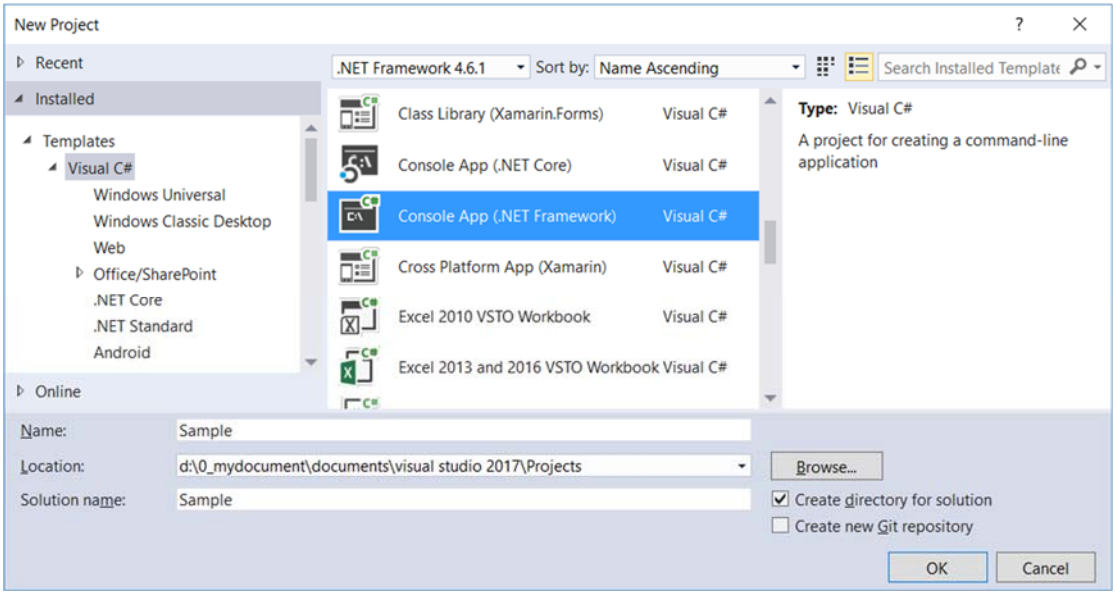
==========

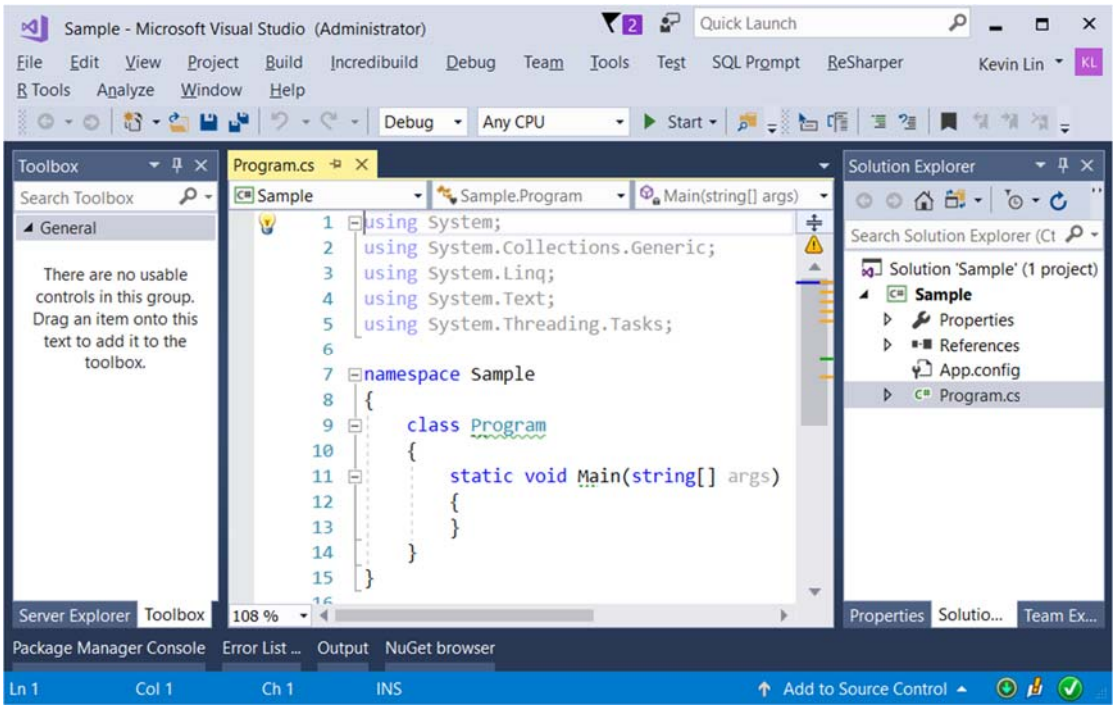
1. New Project

1.1. Create New Project

File --> New --> Project... -->
Visual C# --> **Console App (.Net Framework)** -->
Name: **Sample**


Start Page - Microsoft Visual Studio
File Edit View Project Incredibuild Debug Team Tools Test SQL Prompt ReSharper
New
Open
Start Page
Close
Close Solution
Save Selected Items Ctrl+S
Save Selected Items As...
Save All Ctrl+Shift+S
Page Setup...
Print... Ctrl+P
Account Settings...
Recent Files
Recent Projects and Solutions
Exit Alt+F4
Project...
Web Site... Shift+Alt+N
Repository...
File...
Project From Existing Code...
From Cookiecutter...
Import
Checkout from:
Visual Stud
GitHub
Open Proje


New Project
Recent
Installed
Templates
Visual C#
Windows Universal
Windows Classic Desktop
Web
Office/SharePoint
.NET Core
.NET Standard
Android
Online
.NET Framework 4.6.1
Sort by: Name Ascending
Search Installed Templat
Class Library (Xamarin.Forms) Visual C#
Console App (.NET Core) Visual C#
Console App (.NET Framework) Visual C#
Cross Platform App (Xamarin) Visual C#
Excel 2010 VSTO Workbook Visual C#
Excel 2013 and 2016 VSTO Workbook Visual C#
Type: Visual C#
A project for creating a command-line application
Name: Sample
Location: d:\0_mydocument\documents\visual studio 2017\Projects
Solution name: Sample
Browse...
Create directory for solution
Create new Git repository
OK
Cancel


Sample - Microsoft Visual Studio (Administrator)
Quick Launch
File Edit View Project Build Incredibuild Debug Team Tools Test SQL Prompt ReSharper
R Tools Analyze Window Help
Kevin Lin
Debug
Any CPU
Start
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Program.cs
Sample
Sample.Program
Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Sample
{
class Program
{
static void Main(string[] args)
{
}
}
}
Solution Explorer
Search Solution Explorer (Ct
Solution 'Sample' (1 project)
Sample
Properties
References
App.config
Program.cs
Server Explorer
Toolbox
108 %
Properties
Solutio...
Team Ex...
Package Manager Console
Error List ...
Output
NuGet browser
Ln 1
Col 1
Ch 1
INS
Add to Source Control

==========

2. Program.cs

```
using System;
using System.Text;
namespace Sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 1. System.string ------------------------
            string str1 = "IT";
            str1 += "Handy";
            str1 += "Guy";
            str1 += " Tutorial";
            str1 += " is";
            str1 += " awesome.";
            Console.WriteLine(str1);
            // 2. System.Text.string ------------------------
            StringBuilder strBuilder =
                new StringBuilder("IT");
            strBuilder.Append("Handy");
            strBuilder.Append("Guy");
            strBuilder.Append(" Tutorial");
            strBuilder.Append(" is");
            strBuilder.Append(" awesome.");
            Console.WriteLine(strBuilder.ToString());
            // 3. System.string ------------------------
            string strInt = string.Empty;
            for (int i = 0; i < 100; i++)
            {
                strInt += i + " ";
            }
            Console.WriteLine(strInt);
            // The System.string is manipulated 100 times,
            // thus, it will create 99 orphaned objects
            // until they are garbage collected.
            Console.ReadLine();
        }
    }
}
```

```
ITHandyGuy Tutorial is awesome.
ITHandyGuy Tutorial is awesome.
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 5
9 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77
78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96
 97 98 99
```

```
/*
1.
System.String V.S. System.Text.StringBuilder
-----------------------------------------
1.1.
System.Text.StringBuilder object is mutable
and good for performance,
but System.String is immutable.
Therefore, heavy string manipulation should
use System.Text.StringBuilder.
-----------------------------------------
1.2.
System.String
//System.String
//string str1 = "IT";
//str1 += "Handy";
//str1 += "Guy";
//str1 += " Tutorial";
//str1 += " is";
//str1 += " awesome.";
//Console.WriteLine(str1);
1.2.1.
RAM contains Stack and Heap area.
Stack is for storing object reference variable.
Heap is for storing the object.
* Firstly, str1 pointed "IT"
* Secondly, str1 pointed "ITHandy",
and "IT" became an orphaned object in heap until it is garbage collected.
* Thirdly, str1 pointed "ITHandyGuy",
and "ITHandy" became an orphaned object in heap until it is garbage collected.
* Do so until str1 finally is pointing "ITHandyGuy Tutorial is Awesome.",
and rest of string objects became orphaned objects in heap until they are garbage collected.
    Stack    |   Heap
-------------------------------------------------
             |   "IT"
             |   "ITHandy"
             |   "ITHandyGuy"
             |   "ITHandyGuy Tutorial"
             |   "ITHandyGuy Tutorial is"
 str1   -------> "ITHandyGuy Tutorial is Awesome."
-----------------------------------------
1.3.
System.Text.StringBuilder
//StringBuilder strBuilder =
//    new StringBuilder("IT");
//strBuilder.Append("Handy");
//strBuilder.Append("Guy");
//strBuilder.Append(" Tutorial");
//strBuilder.Append(" is");
//strBuilder.Append(" awesome.");
//Console.WriteLine(strBuilder.ToString());
1.3.1.
No matter how many times the string is manipulated,
StringBuilder always points to the same object instance.
    Stack       |   Heap
```

```
-------------------------------------------------
                |
 StringBuilder ---> "ITHandyGuy Tutorial is Awesome."
*/
```