|  |
| --- |
| SEW |
| Console\_String\_StringBuilder\_Formen |
| sw.ConsoleWriteLine($“{stringbuilder}“); |

|  |
| --- |
| Felix  6.12.2021  1h |

# Inhaltsverzeichnis

Inhalt

[Inhaltsverzeichnis 1](#_Toc89681021)

[Angabe 1 <<…>> 2](#_Toc89681022)

[Theorie 1 <<…>> 2](#_Toc89681023)

[Quellcode 1 <<…>> 2](#_Toc89681024)

[Testfälle 1 <<…>> 3](#_Toc89681025)

# Angabe 1 <<…>>

Öffne den Foliensatz Console\_String\_StringBuilder\_Formen

Teste ob die Ausgaben in der Console auch bei einem StreamWriter funktionieren.

Rufe die Unterschiede zwischen String und StringBuilder in Erinnerung

Implementiere eigene Beispiele schreibe alle Beispiele in eine Datei

Nutze unterschiedliche Arten zum Schreiben in eine Datei.

**Zum Vergleichen**

Equal, String.Compare, variable.Compareto(var)

Vergleiche unterschiedliche Instanzen gleichen Typs mit gleichen Werten

Student s1 = new Student("Kurt");

Student s2 = new Student("Kurt");

-true oder false: s1.Equals(s2)

**Zum Verändern:** Insert, Replace, Remove, PadLeft & PadRight Substring ToUpper ToLower **Zum Suchen:** IndexOf, LastIndexOf, Contains, StartsWith EndsWith

# Theorie 1 <<…>>

Ein String kann nicht verändert werden. Wenn ein String verändert wird, wird eigentlich ein neuer String gebaut, und der alte Verweis zeigt dann auf den neuen String.

Ein StringBuilder kann beliebig oft verändert werden, und ist auch sehr viel schneller als ein string, wenn es ums Verändern geht.

# Quellcode 1 <<…>>

using Console\_String\_StringBuilder\_Formen;

using System.Text;

string text = "some text";

StringBuilder sb = new StringBuilder();

sb.Append("This is a ");

sb.Append("StringBuilder");

sb.Append("I can append text...");

Console.WriteLine($"text is \t {text}");

Console.WriteLine($"StringBuilder is \t {sb}");

using (StreamWriter sw = new StreamWriter("test.txt"))

{

sw.WriteLine($"text is \t {text}");

sw.WriteLine($"StringBuilder is \t {sb}");

}

//zum Vergleichen

Console.WriteLine();

Console.WriteLine("Vergleichen".ToUpper().PadRight(20, '-').PadLeft(30, '-'));

Person p1 = new Person("Kurt");

Person p2 = new Person("Kurt");

Console.WriteLine($"Person.Equals: {p1.Equals(p2)}");

string s1 = "Kurt";

string s2 = "Kurt";

Console.WriteLine($"String.Equals: {s1.Equals(s2)}");

//zum Ersetzen

Console.WriteLine();

Console.WriteLine("Ersetzen".ToUpper().PadRight(20, '-').PadLeft(30, '-'));

Console.WriteLine(s1);

s1 = s1.Replace('K', 'C');

s1 = s1.Replace('u', 'a');

Console.WriteLine(s1);

Console.WriteLine(s2);

s2 = s2.PadLeft(s2.Length + 10, '\*');

s2 = s2.PadRight(s2.Length + 10, '\*');

Console.WriteLine(s2);

Console.WriteLine(s1);

Console.WriteLine(s2);

s1 = s1.ToUpper();

s2 = s2.ToLower();

Console.WriteLine(s1);

Console.WriteLine(s2);

//zum Suchen

Console.WriteLine();

Console.WriteLine("Suchen".ToUpper().PadRight(20, '-').PadLeft(30, '-'));

if (s1.Contains('C'))

Console.WriteLine(s1);

else s1 = s1.Insert(0, "C");

int x = s2.Length + s2.IndexOf('t');

if(x < 8)

Console.WriteLine(x);

else Console.WriteLine(s2.Length);

# Testfälle 1 <<…>>

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung