

▼ C# Fundamentals StringBuilder

StringBuilder:

Uso da classe StringBuilder no .NET

30/03/2017 • 6 minutos para o fim da leitura •   


O objeto `String` é imutável. Sempre que usa um dos métodos na classe `System.String`, você cria um novo objeto de cadeia de caracteres na memória, o que requer uma nova alocação de espaço para esse novo objeto. Em situações em que você precisa realizar repetidas modificações em uma cadeia de caracteres, a sobrecarga associada à criação de um novo objeto `String` pode ser dispendiosa. A classe `System.Text.StringBuilder` pode ser usada quando você deseja modificar uma cadeia de caracteres sem criar um novo objeto. Por exemplo, o uso da classe `StringBuilder` pode melhorar o desempenho ao concatenar várias cadeias de caracteres em um loop.

StringBuilder:

Importando o namespace System.Text

A classe `StringBuilder` é encontrada no namespace `System.Text`. Para evitar ter que fornecer um nome de tipo totalmente qualificado no código, você pode importar o namespace `System.Text`:

C#


 Copiar

```
using System;  
using System.Text;
```

Criando uma instância de um objeto StringBuilder

Você pode criar uma nova instância da classe `StringBuilder` inicializando sua variável com um dos métodos do construtor sobrecarregados, conforme ilustrado no exemplo a seguir.

C#

 Copiar


```
StringBuilder myStringBuilder = new StringBuilder("Hello World!");
```


StringBuilder:

Definindo a capacidade e o comprimento

Embora o `StringBuilder` seja um objeto dinâmico que permite que você expanda o número de caracteres na cadeia de caracteres que ele encapsula, você pode especificar um valor para o número máximo de caracteres que ele pode conter. Esse valor é chamado de capacidade do objeto e não deve ser confundido com o comprimento da cadeia de caracteres que o `StringBuilder` atual contém. Por exemplo, você pode criar uma nova instância da classe `StringBuilder` com a cadeia de caracteres "Hello", que tem um comprimento de 5 caracteres e você pode especificar que o objeto tenha uma capacidade máxima de 25. Quando você modifica o `StringBuilder`, ele não realoca tamanho para si mesmo até que a capacidade seja atingida. Quando isso ocorre, o novo espaço é alocado automaticamente e a capacidade é dobrada. Você pode especificar a capacidade da classe `StringBuilder` usando um dos construtores sobrecarregados. O exemplo a seguir especifica que o objeto `myStringBuilder` pode ser expandido para um máximo de 25 espaços.


C#

 Copiar

```
StringBuilder myStringBuilder = new StringBuilder("Hello World!", 25);
```

Além disso, você pode usar a propriedade `Capacity` de leitura/gravação para definir o comprimento máximo do objeto. O exemplo a seguir usa a propriedade `Capacidade` para definir o comprimento máximo de objeto.

C#

 Copiar

```
myStringBuilder.Capacity = 25;
```

StringBuilder:

Modificando a cadeia de caracteres do StringBuilder

A tabela a seguir lista os métodos que você pode usar para modificar o conteúdo de um `StringBuilder`.

Nome do método	Usar
<code>StringBuilder.Append</code>	Acrescenta informações ao final do <code>StringBuilder</code> atual.
<code>StringBuilder.AppendFormat</code>	Substitui um especificador de formato transmitido em uma cadeia de caracteres com texto formatado.
<code>StringBuilder.Insert</code>	Insere uma cadeia de caracteres ou um objeto no índice especificado do <code>StringBuilder</code> atual.
<code>StringBuilder.Remove</code>	Remove um número especificado de caracteres do <code>StringBuilder</code> atual.
<code>StringBuilder.Replace</code>	Substitui todas as ocorrências de um caractere ou cadeia de caracteres especificada no <code>StringBuilder</code> atual por outro caractere ou cadeia de caracteres especificada.

StringBuilder:

Acrescentar

O método **Append** pode ser usado para adicionar texto ou uma representação de cadeia de caracteres de um objeto ao final de uma cadeia de caracteres representada pelo **StringBuilder** atual. O exemplo a seguir inicializa um **StringBuilder** para "Hello World" e, em seguida, acrescenta algum texto ao final do objeto. Espaço é alocado automaticamente conforme necessário.

C#

 Copiar


```
StringBuilder myStringBuilder = new StringBuilder("Hello World!");  
myStringBuilder.Append(" What a beautiful day.");  
Console.WriteLine(myStringBuilder);  
// The example displays the following output:  
//      Hello World! What a beautiful day.
```

StringBuilder:

AppendFormat

O método [StringBuilder.AppendFormat](#) adiciona texto ao final do objeto [StringBuilder](#). Ele dá suporte ao recurso de formatação de composição (para obter mais informações, confira [Formatação de composição](#)) chamando a implementação [IFormattable](#) do objeto ou objetos a serem formatados. Portanto, aceita as cadeias de caracteres de formato padrão para valores numéricos, data e hora e enumeração, cadeias de caracteres de formato personalizado para valores numéricos e de data e hora e cadeias de caracteres de formato definidas para tipos personalizados. (Para obter informações sobre formatação, consulte [tipos de formatação](#).) Você pode usar esse método para personalizar o formato de variáveis e acrescentar esses valores a um [StringBuilder](#). O exemplo a seguir usa o método [AppendFormat](#) para colocar um valor inteiro, formatado como um valor de moeda, no final de um objeto [StringBuilder](#).

C#

 Copiar

```
int MyInt = 25;
StringBuilder myStringBuilder = new StringBuilder("Your total is ");
myStringBuilder.AppendFormat("{0:C} ", MyInt);
Console.WriteLine(myStringBuilder);
// The example displays the following output:
//      Your total is $25.00
```

StringBuilder:

Inserir

O método `Insert` adiciona uma cadeia de caracteres ou um objeto a uma posição especificada no objeto `StringBuilder` atual. O exemplo a seguir usa esse método para inserir uma palavra na sexta posição de um objeto `StringBuilder`.

C#

 Copiar

```
StringBuilder myStringBuilder = new StringBuilder("Hello World!");  
myStringBuilder.Insert(6,"Beautiful ");  
Console.WriteLine(myStringBuilder);  
// The example displays the following output:  
//      Hello Beautiful World!
```


StringBuilder:

Remove

Você pode usar o método `Remove` para remover um número especificado de caracteres do objeto `StringBuilder` atual, começando em um índice com base zero especificado. O exemplo a seguir usa o método `Remove` para reduzir um objeto `StringBuilder`.

C#

 Copiar

```
StringBuilder myStringBuilder = new StringBuilder("Hello World!");  
myStringBuilder.Remove(5,7);  
Console.WriteLine(myStringBuilder);  
// The example displays the following output:  
//      Hello
```

StringBuilder:

Substitua

O método `Replace` pode ser usado para substituir caracteres dentro do objeto `StringBuilder` com outro caractere especificado. O exemplo a seguir usa o método `Replace` para pesquisar, em um objeto `StringBuilder`, todas as instâncias do caractere de ponto de exclamação (!) e substituí-los com o caractere de ponto de interrogação (?).

C#

 Copiar

```
StringBuilder myStringBuilder = new StringBuilder("Hello World!");  
myStringBuilder.Replace('!', '?');  
Console.WriteLine(myStringBuilder);  
// The example displays the following output:  
//      Hello World?
```


StringBuilder:

Convertendo um objeto StringBuilder em uma cadeia de caracteres

Você deve converter o objeto `StringBuilder` em um objeto `String` para transmitir a cadeia de caracteres representada pelo objeto `StringBuilder` para um método que tem um parâmetro `String` ou exibi-lo na interface do usuário. Você faz essa conversão chamando o método `StringBuilder.ToString`. O exemplo a seguir chama vários métodos `StringBuilder` e chama o método `StringBuilder.ToString()` para exibir a cadeia de caracteres.

StringBuilder:

C#

 Copiar

```
using System;
using System.Text;

public class Example
{
    public static void Main()
    {
        StringBuilder sb = new StringBuilder();
        bool flag = true;
        string[] spellings = { "recieve", "receeve", "receive" };
        sb.AppendFormat("Which of the following spellings is {0}:", flag);
        sb.AppendLine();
        for (int ctr = 0; ctr <= spellings.GetUpperBound(0); ctr++) {
            sb.AppendFormat("    {0}. {1}", ctr, spellings[ctr]);
            sb.AppendLine();
        }
        sb.AppendLine();
        Console.WriteLine(sb.ToString());
    }
}

// The example displays the following output:
//      Which of the following spellings is True:
//          0. recieve
//          1. receeve
//          2. receive
```

StringBuilder:

- **Links de documentação oficial:**
- **Documentação:**
- <https://docs.microsoft.com/pt-br/dotnet/standard/base-types/stringbuilder>



Dúvidas?