



PROGRAMAÇÃO O.O. (C#)



Strings em C#
Professor: João Luiz Lagôas



Strings em C#



- Como formulários trabalham com frequência com dados textuais, tiraremos esse momento para nos aprofundar em alguns detalhes pertinentes ao uso de strings na linguagem C#.
- String é um tipo de dado muito comum em linguagens de programação. Ao declararmos uma variável desse tipo, estamos reservando um espaço de memória capaz de armazenar uma informação textual.
- Diferente de C, onde a criação de uma string pode se tornar um trabalho mais técnico (envolve alocação de memória com malloc, criação de vetores, etc), em C# a criação de uma string acontece de forma natural. Observe o exemplo:

```
string pokemon = "Pikachu";
```



Strings em C#



- Dentro do namespace System, há uma **classe** denominada **String** e, assim como já aprendemos, para criarmos **objetos** de uma determinada classe precisamos usar a palavra reservada de instânciação **new** em conjunto com um **construtor**.

```
FuncaoLinear f1 = new FuncaoLinear(2,5);
```

Objeto de Função Linear é criado!

Strings em C#



Mas no caso da classe **string**, não chamamos nenhum construtor na criação de um objeto string! Cadê o new e o construtor?!?

```
string pokemon = "Pikachu";
```

???



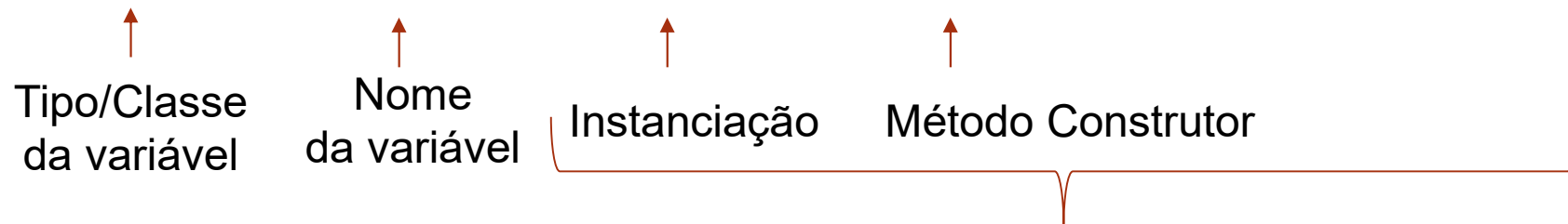
Criando objetos da classe String



- O código abaixo é uma declaração onde estamos construindo um objeto string que armazena o valor “pikachu” e estamos o atribuindo à variável pokemon.

```
char[] vetorDeCaracteres = {'p', 'i', 'k', 'a', 'c', 'h', 'u'};
```

```
string pokemon = new string(vetorDeCaracteres);
```



Objeto "Pikachu" criado!

- Ele é equivalente à declaração:

```
string pokemon = "Pikachu";
```

Criando objetos da classe String

- ❖ Entendendo o passo a passo.



```
string pokemon = new string(vetorDeCaracteres);
```

Criando objetos da classe String

- ❖ Entendendo o passo a passo.



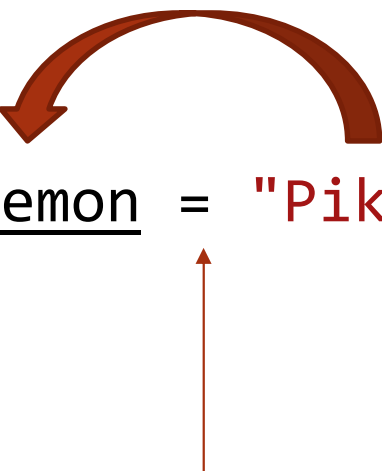
```
string pokemon = "Pikachu";
```



Objeto criado!

Criando objetos da classe String

- ❖ Entendendo o passo a passo.



```
string pokemon = "Pikachu";
```

Objeto atribuído à variável

Criando objetos da classe String



- Como informações textuais (strings) são muito utilizadas em programas, a plataforma .NET permite a criação de objetos do tipo string de uma maneira bem fácil.

```
{ char[] vetorDeCaracteres = {'p', 'i', 'k', 'a', 'c', 'h', 'u'};  
  string pokemon = new string(vetorDeCaracteres);
```

```
{ string pokemon = "Pikachu";
```

- As duas declarações acima são equivalentes, fazem exatamente a mesma coisa! Enquanto na primeira eu estou chamando explicitamente o construtor de string para criar um objeto (deixando claro para nós que **string** é uma classe), na segunda eu simplesmente declaro o objeto string que gostaria de criar.
- Em outras palavras, você não precisa chamar o **new** nem a o construtor pois, implicitamente, ao se declarar “Pikachu”, tudo isso aconteceu e o objeto foi criado automaticamente.

A Classe String



- Agora que entendemos que uma String é uma classe e que informações textuais entre aspas são objetos (ex: "**Pikachu**"), a próxima pergunta natural que poderíamos fazer é:
- Será que há atributos e métodos que eu como programador posso utilizar dessa classe!? Afinal de contas, já entendemos também que classes são compostas por atributos e métodos. Não só podemos criar as nossas como também podemos utilizar classes prontas que ficam guardadas em algum namespace.

Atributos e Métodos da Classe String



Nome do Atributo	Funcionamento
<code>public int Length</code>	Retorna um inteiro indicando a quantidade de caracteres no objeto string.

Assinatura do Método	Funcionamento
<code>public bool Contains (string value);</code>	Retorna true ou false indicando se uma determinada string existe dentro da string chamadora do método.
<code>public string Replace</code> <code>(char oldChar, char newChar);</code> <code>public string Replace</code> <code>(string oldSubString,</code> <code>string newSubString);</code>	Retorna uma nova cadeia de caracteres na qual todas as ocorrências de um caractere Unicode ou string especificada na instância chamadora são substituídas por outro caractere Unicode ou outra string especificada.
<code>public string Trim ();</code>	Remove todos os caracteres de espaço em branco à esquerda e à direita do objeto atual, produzindo uma nova string.

Atributos e Métodos da Classe String



Nome do Atributo	Funcionamento
<code>public int Length</code>	Retorna um inteiro indicando a quantidade de caracteres no objeto string.

Assinatura do Método	Funcionamento
<code>public bool Contains (string value);</code>	Retorna true ou false indicando se uma determinada string existe dentro da string chamadora do método.
<code>public string Replace</code> <code>(char oldChar, char newChar);</code> <code>public string Replace</code> <code>(string oldSubString,</code> <code>string newSubString);</code>	Retorna uma nova cadeia de caracteres na qual todas as ocorrências de um caractere ou string especificada na primeira string são substituídas por outra string.
<code>public string Trim ();</code>	Retorna uma nova string com os caracteres de espaço em branco removidos da esquerda e à direita do objeto atual, produzindo uma nova string.

Vejamos alguns exemplos de uso do atributo e desses métodos!

Atributo Length

❖ Exemplo



```
string str = "abcdefg";
```

```
Console.WriteLine("1) O tamanho de '{0}' é {1}", str,  
str.Length);
```

```
Console.WriteLine("2) O tamanho de '{0}' é {1}", "xyz",  
"xyz".Length);
```

```
int length = str.Length;
```

```
Console.WriteLine("3) O tamanho de '{0}' é {1}", str, length);
```

```
// O exemplo exibirá a seguinte saída:
```

```
// 1) O tamanho de 'abcdefg' é 7
```

```
// 2) O tamanho de 'xyz' é 3
```

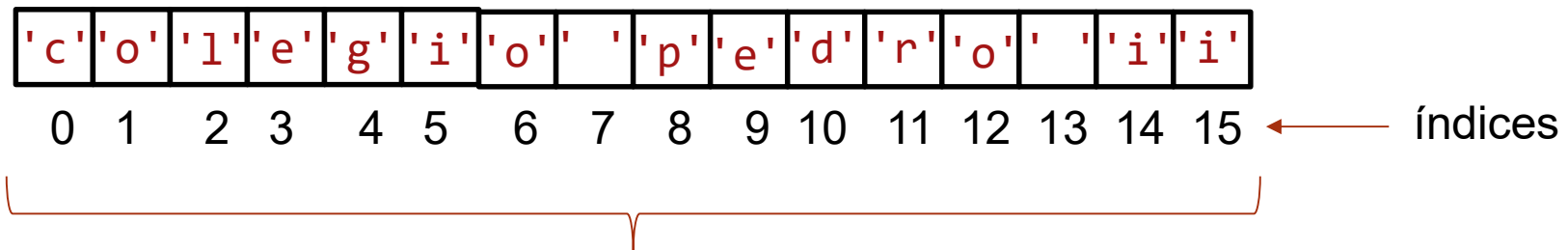
```
// 3) O tamanho de 'abcdefg' é 7
```

Atributo Length



- Apesar de String em C# ser considerada uma classe de onde produzimos objetos, internamente, ela é organizada como um vetor de caracteres.

```
string str = "colégio pedro ii";
```



str.Length vale 16

- Sendo assim, se for necessário percorrer um objeto de string str, basta criar um loop onde a variável de índice i irá variar entre 0 e str.Length-1.
- A cada iteração do loop, você terá acesso a um caractere da string, podendo fazer comparações, por exemplo.

Atributo Length

❖ Exemplo



```
string str = "colégio pedro ii";  
int numEspaços = 0;  
  
for(int i = 0; i < str.Length; i++)  
{  
    if (str[i] == ' ')  
        numEspaços++;  
}
```

```
Console.WriteLine("A quantidade de espaços na string  
{0} é {1}", str, numEspaços);
```

```
// Esse exemplo produz a seguinte saída:  
// A quantidade de espaços na string colégio pedro ii é 2
```

Método Contains

❖ Exemplo



```
string s1 = "Google Chorme é melhor que Internet Edge";  
string s2 = "Internet";  
bool b = s1.Contains(s2);  
Console.WriteLine("'{0}' está na string '{1}': {2}", s2, s1, b);
```

```
// Esse exemplo produz a seguinte saída:  
// 'Internet' está na string 'Google Chrome é melhor  
que Internet Edge': True
```


Método Replace

❖ Exemplo



```
String str = "1 2 3 4 5 6 7 8 9";  
Console.WriteLine("String Original: \"{0}\"", str);  
Console.WriteLine("String Nova: \"{0}\"", str.Replace(' ',  
' , '));
```

```
// Esse exemplo produz a seguinte saída:  
// String Original: "1 2 3 4 5 6 7 8 9"  
// String Nova: "1,2,3,4,5,6,7,8,9"
```

Método Replace

❖ Exemplo



```
string errString = "This docment uses 3 other docments to  
docment the documentation";  
  
Console.WriteLine("A string original é:{0}'{1}'{0}",  
Environment.NewLine, errString);  
  
string correctString = errString.Replace("docment", "document");  
  
Console.WriteLine("Depois de corrigirmos a string, teremos:  
{0}'{1}'", Environment.NewLine, correctString);
```

```
// Esse exemplo produz a seguinte saída:  
// A string original é:  
// 'This docment uses 3 other docments to docment the  
documentation'  
// Depois de corrigirmos a string, teremos:  
// 'This document uses 3 other documents to document the  
documentation'
```

Método Trim

❖ Exemplo



```
Console.Write("Entre com seu primeiro nome: ");

string firstName = Console.ReadLine();

Console.Write("Entre com seu nome do meio: ");

string middleName = Console.ReadLine();

Console.Write("Entre com seu último nome: ");

string lastName = Console.ReadLine();

Console.WriteLine();

Console.WriteLine("Você digitou '{0}', '{1}', and '{2}'.", firstName, middleName,
lastName);

string name = ((firstName.Trim() + " " + middleName.Trim()).Trim() + " " +
lastName.Trim()).Trim();

Console.WriteLine("O resultado é " + name + ".");
```

Método Trim

❖ Exemplo



Considere que o usuário ao digitar, irá inserir espaços em branco antes e depois do próprio nome.

```
// Esse exemplo produz a seguinte saída:  
// Entre com seu primeiro nome: Steve  
// Entre com seu nome do meio: Grant  
// Entre com seu ultimo nome: Rogers  
// Você digitou ' Steve ', ' Grant ', and ' Rogers'.  
// O resultado é Steve Grant Rogers.
```

A Classe String

❖ Exemplo

- Para quem tiver curiosidade, na documentação oficial das classes presentes na plataforma .NET, você encontrará todos os atributos e métodos que essa classe disponibiliza!
- Link para a documentação: <https://docs.microsoft.com/pt-br/dotnet/api/system.string?view=netframework-4.8>
- Não somente, todos os métodos acompanham exemplos para facilitar o entendimento de seus funcionamentos.

