**Boas práticas de programação em Delphi #2 – Parâmetros de métodos**

Publicado em [**30 de janeiro de 2014**](http://blog.tecsystem.com.br/index.php/boas-praticas-de-programacao-em-delphi-2-parametros-de-metodos/) por [**Breno Martinusso**](http://blog.tecsystem.com.br/index.php/author/breno/)

É importante conhecer e classificar corretamente os parâmetros de um método (seja uma function ou procedure). Em Object Pascal existem 4 tipos de classificações possíveis:value, variable, constant e out.

**value**

Parâmetros do tipo **value** (o padrão) são passados ​​por valor. Isso significa que ao passar uma variável, o método cria uma cópia da mesma permitindo a alteração do parâmetro, porém sem interferir na variável original.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **var**  S: **string**;  **begin**  S := 'Tecsystem';  FazAlgo(S);  ShowMessage(S); *// S = 'Tecsystem'*  **end**;    **procedure** FazAlgo(S: **string**); *// S é um parâmetro value*  **begin**  S := 'Tecsystem Tecnologia em Software';  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**; |

**variable**

Parâmetros **variable** (var) ​​são passados ​​por referência. Esse tipo de parâmetro atua com um ponteiro ao invés de uma cópia. Desse modo as alterações feitas no parâmetro dentro do corpo de um método interfere na variável original.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **var**  S: **string**;  **begin**  S := 'Tecsystem';  FazAlgo(S);  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**;    **procedure** FazAlgo(**var** S: **string**); *// S é um parâmetro variable*  **begin**  S := 'Tecsystem Tecnologia em Software';  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**; |

**constant**

Um parâmetro **constant** (const) funciona como uma constante local ou uma variável somente leitura. São semelhantes aos parâmetros value, exceto que não é possível atribuir um valor ao parâmetro dentro do escopo do método.

O uso de um parâmetro constant melhora ligeiramente o desempenho, torna mais legível a operação e impede alterações acidentais no valor do parâmetro.

Ao tentar alterar o valor do parâmetro, é gerado um erro ("[DCC Error] E2064 Left side cannot be assigned to") em tempo de compilação .

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12 | **var**  S: **string**;  **begin**  S := 'Tecsystem';  FazAlgo(S);  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**;    **procedure** FazAlgo(**const** S: **string**); *// S é um parâmetro constant*  **begin**  S := 'Tecsystem Tecnologia em Software'; *// [DCC Error] E2064 Left side cannot be assigned to*  **end**; |

Mas é importante ressaltar que se passado um uma referência de objeto como parâmetro constant, é possível modificar as propriedades do objeto.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | **var**  Empresa: TEmpresa;  **begin**  Empresa := TEmpresa.Create();  Empresa.Nome := 'Tecsystem';    FazAlgo(Empresa);  ShowMessage(Empresa.Nome); *// Empresa.Nome = 'Tecsystem Tecnologia em Software'*  **end**;    **procedure** TForm2.FazAlgo(**const** Empresa: TEmpresa);  **begin**  Empresa.Nome := 'Tecsystem Tecnologia em Software';  ShowMessage(Empresa.Nome); *// Empresa.Nome = 'Tecsystem Tecnologia em Software'*  **end**; |

**out**

O parâmetro **out** (out), assim como o parâmetro variável, é passado por referência. Com um parâmetro out, no entanto, o valor inicial da variável referenciada é descartada pelo método. Ou seja, o parâmetro out funciona apenas como output, não permitindo fornecer nenhum valor de entrada.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **var**  S: **string**;  **begin**  S := 'Tecsystem';  FazAlgo(S);  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**;    **procedure** FazAlgo(out S: **string**); *// S é um parâmetro out*  **begin**  ShowMessage(S); *// S = ''*  S := 'Tecsystem Tecnologia em Software';  ShowMessage(S); *// S = 'Tecsystem Tecnologia em Software'*  **end**; |