Nomenclatura Geral

Nomenclatura

Componentes

Os componentes possuem seus prefixos utilizando letras que remetem ao nome completo da classe do componente. Componentes herdados dos componentes listados e sem conexão ao banco de dados devem possuir o mesmo prefixo do componente principal, com exceção dos componentes visuais com conexão ao banco de dados, que utilizam o prefixo “DB“ seguido do padrão identificado logo abaixo. Exemplo: DBED_Nome é um componente da classe TDBEdit, responsável pela representação de um nome.

Seguem alguns exemplos de prefixos padrões utilizados:

ED_: TEdit;	TS_: TTabSheet;
PN_: TPanel;	RB_: TRadioButton;
GB_: TGroupBox;	OD_: TOpenDialog;
LB_: TLabel;	SD_: TSaveDialog;
MM_: TMemo;	Q_: TQuery, TFDQuery;
BT_: TButton (Botões num geral);	DS_: TDataSource;
CK_: TCheckBox;	CDS_: TClientDataSet;
CB_: TComboBox;	DSP_: TDataSetProvider;
LT_: TListBox;	T_: TTable;
RG_: TRadioGroup;	FR_: TFrxReport;
PC_: TPageControl;	FDS_: TFrxDBDataSet;
TB_: TToolBar;	RV_: TRvProject;

GD_: TGrid;	DSC_: TRvDataSetConnection;
MI_: TMenuItem;	SH_: TShape
IM_: TImage	TM_: TTimer
NB_: TNotebook	Page: TPage

Exemplo de componentes visuais com conexão ao banco de dados:

DBED_: TDBEdit;	DBGD_: TDBGrid;
DBCK_: TDBCheckBox;	DBRG_: TDRadioGroup;

Variáveis

O nome da variável deve descrever de maneira clara e sem abreviações a função da variável.
Utilizar a notação PascalCase para o nome da variável.
As variáveis podem ter ainda um prefixo para adicionar mais informações sobre sua origem.
Para **variáveis locais** utiliza-se o prefixo **‘L’**, para **argumentos (Parâmetros) de métodos** utiliza-se o **'A’**. Para **atributos (Field)** de classes utiliza-se o **‘F’**.

```
type
  TPessoa = class
    protected
      FNome: String; // Campo nome da classe Pessoa. Prefixo F
    public
      constructor Create(ANome: String); // Argumento do método. Prefixo A.
  end;

implementation

constructor TPessoa.Create(ANome: String);
var
  LNome: String; // Variável local. Prefixo L
begin

end;
```

Constantes

Para nomes de constantes, o padrão utilizado é:

```
const
  NOME_DA_CONSTANTE_1 = 0;
  NOME_DA_CONSTANTE_2 = False;
  NOME_DA_CONSTANTE_3 = ‘Teste’;
```

O tipo da constante é deduzido através do valor dessa constante no momento da declaração. Porém, é possível declarar explicitamente um tipo para as constantes, e em alguns casos, como o de arrays, é necessário que esse tipo seja declarado.

```
const
  NOME_DA_CONSTANTE_4: Double = 10;
  NOME_DA_CONSTANTE_5: array [1..2] of String = (‘a’, ‘b’);
```

Métodos

Assim como o nome das variáveis, o nome do método deve seguir o mesmo modelo, com nomes que descrevem a função do método, utilizando também a notação PascalCase.
Ao editar um método, é importante manter sua função original, caso sua funcionalidade seja alterada, deve-se refatorar o seu nome para que permaneça fiel ao seu comportamento.

Exemplo 1: Criar um método que faça a confirmação de um rotina de gravação. Sem retorno e sem argumentos.

```
procedure ConfirmarGravacao();
```

Exemplo 2: Criar um método que faça o cálculo da área de um retângulo. Seu retorno será um inteiro, esse método terá ainda dois argumentos, altura e largura.

```
function CalcularArea(AAltura, ALargura: Integer): Integer;
```

Units

As units padrões utilizadas anteriormente para as entidades eram nomeadas acrescentando 1, 2, 3 ou 4. Para facilitar o entendimento na leitura do nome da unit, elas passam a ser utilizadas no singular, seguido do nome de sua funcionalidade. Por exemplo, a entidade “Empresa” passa a ser utilizada como:
● Empresas1: Empresa.Manutencao;
● Empresas2: Empresa.Cadastro;
● Empresas3: Empresa.Pesquisa;
● Empresas4: Empresa.DataModule;
Para units que tenham outra função que não se enquadrem nessas descritas anteriormente, o padrão é utilizar o nome da entidade seguido da descrição de sua função.
Exemplo: Empresa.Copiar.

Classes

O padrão Delphi para nome de classes aconselha que ela inicie com o prefixo T (*Type*). É aconselhável que o nome dessas classes sejam descritivos quanto à sua entidade, função e/ou tipo. Por exemplo, uma classe que define a entidade produtos poderia ter o nome *TProdutoModelo*, enquanto o formulário para importação desses produtos poderia ser *TProdutoImportacaoFormulario.*

Revision #28
Created 21 December 2020 17:16:29
Updated 7 July 2026 17:41:51 by Carlos Filho