

Nesta seção você encontra artigos intermediários sobre Delphi Win32 e Delphi .NET

ClientDataSet

Automatizando o tratamento de Erros

Quem já criou algum tipo de aplicativo que fizesse acesso com banco de dados e que teve que implementar alguma forma para tratar os erros retornados pelo SGBD, sabe o quanto vai ser útil a utilização desse componente (criado neste artigo), pois com ele não será mais necessário ter que implementar algum método de tratamento em todos os componentes ClientDataSet espalhados pelo aplicativo.

O principal objetivo desse componente é retirar do desenvolvedor a necessidade de implementar rotinas para tratar os erros retornados no banco de dados através dos eventos OnReconcileError e OnPostError. Nesse artigo irei criar esse novo componente derivado da classe TClientDataSet. Os erros tratados pelo componente são baseados nas mensagens retornadas utilizando o banco de dados Firebird.

Criando o componente MyClientDataset

Vamos iniciar a criação de componente, para isso inicie o Delphi (estou utilizando a versão 7, mas nada impede que se use outra versão), no menu principal, entre na opção Component >> New Component. Com isso será aberta a tela da **Figura 1**. Veja a **Tabela 1**.

Após informar os valores acima, clique em OK, será criado o arquivo fonte do nosso componente. Agora só precisamos implementar as novas funcionalidades no componente. Vamos iniciar incluindo no uses do nosso componente as unit's necessárias para compilação e criar um novo tipo chamado TMyErrors para identificar os erros retornados (**Listagem 1**).

Com isso concluído, vamos inserir os métodos, o construtor e o destrutor do componente, irei criar duas funções sobrecarregadas com o nome de DetectarErros para identificar o erro retornado pelo SGBD, ver em qual opção do nosso



Rodrigo Lazoti

(rodrigolazoti@yahoo.com.br)

é programador e desenvolvedor Delphi, .Net, Java, Php e Asp. Possui certificação SCJP e atualmente trabalha como consultor J2EE.

tipo criado a mensagem de erro se enquadra e retorná-la como resultado da função. Criei também dois procedimentos sobrecarregados com o nome de RetornarErros que tem como finalidade disparar uma nova exceção tratada para o aplicativo. E para finalizar criei dois métodos para fazer o vínculo com os eventos onReconcileError e onPostError do componente. Primeiro vamos incluir as declarações da sessão private do componente (**Listagem 2**).

Antes de criarmos o corpo desses métodos vamos incluir os métodos restantes para criar o corpo de todos os métodos de uma vez. Veja na **Listagem 3** os métodos da sessão *public*.

Pronto, agora já temos todos os métodos que iremos usar em nosso componente declarados. Aperte Ctrl+Shift+C para que seja criado automaticamente o corpo de todos os métodos do compo-

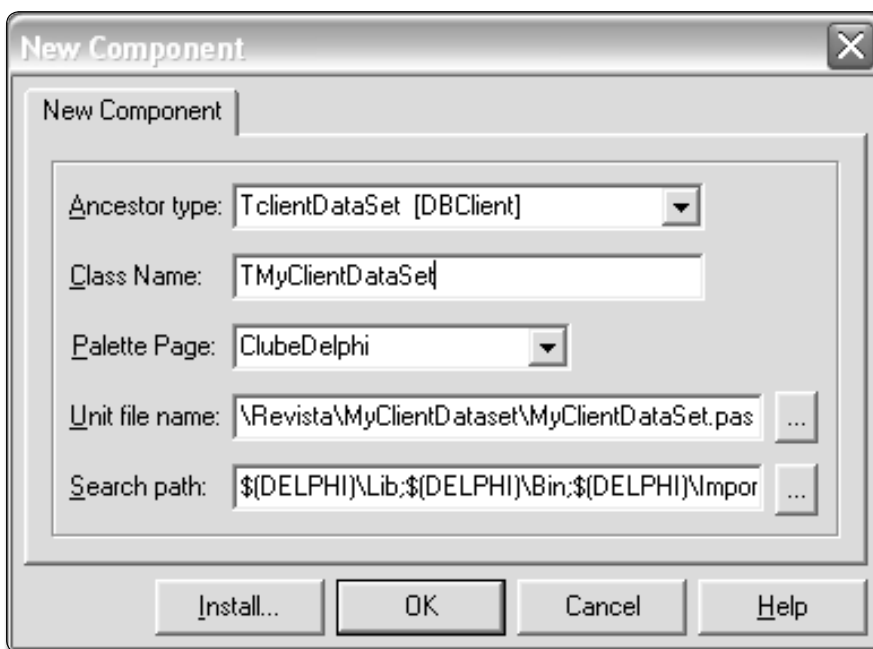


Figura 1. Criando um novo componente derivado da classe TClientDataSet

esteja preparado para o futuro!

treinamentos profissionais em TI

CobIT ITIL

Material Didático em Português • Turmas Reduzidas • Instrutores Certificados
COMPLETA INFRA-ESTRUTURA • FÁCIL LOCALIZAÇÃO • COFFEE-BRAKE • ESTACIONAMENTO CONVENIADO

Faça Tecnologia em Sistema de Informação na FMG: Formamos profissionais para as Maiores Empresas do Brasil

Contribuindo para o aperfeiçoamento profissional dos profissionais de TI, a FMG IT Learning, através de parcerias oficiais, promove os Treinamentos ITIL Foundations, CobIT, SOX, PMI e ISO.

O objetivo destes programas de formação é dotar o gestor de conhecimentos implícitos na metodologia das melhores práticas de gestão, com conteúdo e estudos reais de caso, especificamente elaborada para a área de TI.

fmg
IT Learning

SOLICITE INFORMAÇÕES SOBRE NOVAS TURMAS

rua general glicério, 45 • 3º andar • cep: 09015-190 • santo andré, são paulo – brasil
 informações: 11 4427.4687 | www.fmgnet.com.br | cursos@fmgnet.com.br

www.fmgnet.com.br

Campo	Valor	Descrição
Ancestor type	TClientDataSet	Contém a classe que nosso componente será derivado.
Class name	TMyClientDataSet	O nome da classe do novo componente
Palette Page	ClubeDelphi	O nome da aba onde o componente será instalado.
Unit file name	C:\Arquivos de programas\Borland\Delphi7\Lib\MyclientDataSet.pas	Nome e local do arquivo fonte do componente criado.

Tabela 1. Informações iniciais para criação do componente

Listagem 1. Declarando o novo tipo de retorno e incluindo as unit's no uses

```
unit MyClientDataSet;

interface

uses
  SysUtils, Classes, DB, DBClient;

type
  TMyErrors = (meViolacaoChave, meValidacao, meChavePrimaria, meChaveEstrangeira,
  meConflito, meOutros);

type
  TMyClientDataSet = class(TClientDataSet)
  private
    { Private declarations }
  protected
    { Protected declarations }
  public
    { Public declarations }
  published
    { Published declarations }
  end;

procedure Register;

implementation

procedure Register;
begin
  RegisterComponents('ClubeDelphi', [TMyClientDataSet]);
end;

end.
```

Listagem 2. Declarando novos métodos no componente

```
private
{ Private declarations }
FTratarErros :Boolean;
function DetectarErros(e: EReconcileError): TMyErrors; overload;
function DetectarErros(e: EDatabaseError): TMyErrors; overload;
procedure RetornarErros(cds: TClientDataSet; Err :EReconcileError); overload;
procedure RetornarErros(cds: TClientDataSet; Err :EDatabaseError); overload;
```

Listagem 3. Declarando novos métodos no componente

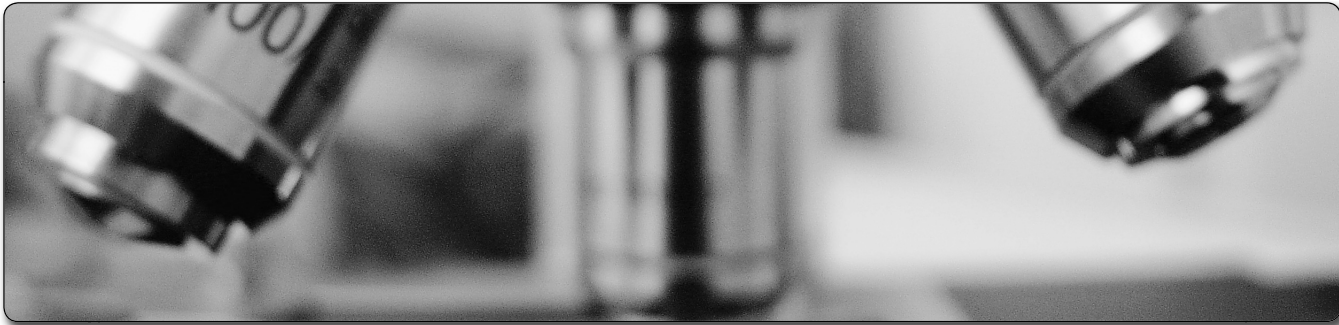
```
public
{ Public declarations }
constructor Create(aOwner :TComponent); override;
destructor Destroy; override;
procedure MyReconcileError(DataSet: TCustomClientDataSet; E: EReconcileError;
UpdateKind: TUpdateKind; var Action: TReconcileAction);
procedure MyPostError(DataSet: TDataSet; E: EDatabaseError;
var Action: TDataAction);
```

nente e insira a codificação dos métodos conforme a **Listagem 4**.

O código está todo comentado para fácil entendimento. Pronto, nosso componente está finalizado, salve a unit. No menu principal do Delphi clique em Component >> Install Component. Na opção Unit file name informe o local e nome da unit criada, clique em OK, será aberta uma janela com o pacote DCLUSR.DPK, compile o pacote, com isso o novo componente já estará disponível para o uso. Experimente utilizar o novo componente com os componentes do dbExpress acessando o Firebird, e veja que erros do servidor, como violação de chave, serão automaticamente tratados.

Conclusão

Como vimos nesse artigo, a criação de componentes pode nos ajudar a não escrever códigos repetitivos, tornando a criação a manutenção dos aplicativos mais fácil. ●



Listagem 4. Codificando os métodos do componente

```

constructor TMyClientDataSet.Create(aOwner: TComponent);
begin
  // Associa os eventos aos métodos
  inherited Create(aOwner);
  Self.OnPostError := MyPostError;
  Self.OnReconcileError := MyReconcileError;
end;
destructor TMyClientDataSet.Destroy;
begin
  inherited Destroy;
end;
function TMyClientDataSet.DetectarErros(e: EReconcileError): TMyErrors;
begin
  // Verifica qual o erro ocorrido pela mensagem
  if (Pos('VALIDATION ERROR', UpperCase(E.Message)) < 0) then
    Result := meValidacao
  else if (Pos('VIOLATION OF FOREIGN KEY', UpperCase(E.Message)) < 0) then
    Result := meChaveEstrangeira
  else if (Pos('VIOLATION OF PRIMARY OR UNIQUE KEY',
    UpperCase(E.Message)) < 0) then
    Result := meChavePrimaria
  else if (Pos('RECORD NOT FOUND OR CHANGED BY ANOTHER USER',
    UpperCase(E.Message)) < 0) then
    Result := meConflito
  else
    Result := meOutros;
end;
function TMyClientDataSet.DetectarErros(e: EDatabaseError): TMyErrors;
begin
  // Trata Key Violation
  if (Pos('KEY VIOLATION', UpperCase(E.Message)) < 0) then
    Result := meViolacaoChave
  else
    Result := meOutros;
end;
procedure TMyClientDataSet.RetornarErros(cds: TClientDataSet;
  Err: EReconcileError);
begin
  // Faz o tratamento do OnReconcileError
  // Cancela o Update
  cds.CancelUpdates;
  cds.Refresh;
  // Trata o tipo de erro
  case DetectarErros(Err) of
    meValidacao:
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' + #10#13+

```

```

        'Valor inválido ou nulo foi encontrado em campos que contém restrições. ' +
        'Motivo: ' + Err.Message);
    meChavePrimaria:
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' + #10#13+
        'O registro não pode ser gravado, ocorreu um erro de duplicidade na chave primária. ' +
        'Motivo: ' + Err.Message);
    meChaveEstrangeira:
      raise Exception.Create('Erro ao efetuar operação ' + cds.Name +
        '.' + #10#13+ 'Dependência inválida entre tabelas ligadas. ' +
        'Motivo: ' + Err.Message);
    meConflito:
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' + #10#13+
        'O registro foi modificado ou deletado por outro usuário. ' +
        'Motivo: ' + Err.Message);
    else
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' + #10#13+
        'Erro desconhecido: ' + Err.Message);
  end;
end;

procedure TMyClientDataSet.RetornarErros(cds: TClientDataSet;
  Err: EDatabaseError);
begin
  cds.CancelUpdates;
  cds.Refresh;
  case DetectarErros(Err) of
    meViolacaoChave:
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' +
        'Já existe um registro com o código informado!');
    else
      raise Exception.Create('Erro ao salvar ' + cds.Name + '.' + #10#13+
        'Erro desconhecido: ' + Err.Message);
  end;
end;
procedure TMyClientDataSet.MyPostError(DataSet: TDataSet;
  E: EDatabaseError; var Action: TDataAction);
begin
  RetornarErros(Self, E);
end;
procedure TMyClientDataSet.MyReconcileError(
  DataSet: TCustomClientDataSet;
  E: EReconcileError; UpdateKind: TUpdateKind;
  var Action: TReconcileAction);
begin
  RetornarErros(Self, E);
end;

```

Impressão Rápida em Matriciais...**RDprint 4.0**

O mais completo componente para impressão em MATRICIAIS !
LIDERANÇA absoluta na sua categoria !

Ideal para Notas Fiscais, Duplicatas, Boletos Bancários, etiquetas e relatórios em geral.

- Opção para impressão colorida
- Ajustes de margens para impressão gráfica
- Opção para ocultar a barra de progresso
- Variáveis PAGINAS, DATA, HORA e TÍTULO

Novo form de SETUP com :

- Mapeamento das impressoras e Modelos
- Seleção de páginas igual ao word (1-5,7,8)
- Opção para Inverter e Agrupar cópias na impressão

Novo form de PREVIEW com:

- Função para Procura de TEXTO no relatório
- ROLAGEM com salto automático de página
- ARRASTO da imagem do preview
- StatusBar com informações da impressão
- Novos ícones personalizados

- * Disponível para Delphi 5, 6, 7, 2005 e 2006 (VCL)
- * Compatível com todas as versões do Windows
- * Imprime em portas LPT / COM e USB (modo gráfico)



Fone/Fax (14) 3454-7880
www.deltress.com.br