[Insira um comentário] Última atualização : 08/09/04

Acessando o Firebird via Delphi 8 .NET

Firebird

O Firebird é um banco de dados relacional "Open-Source" bastante difundido meio a comunidade de programadores no Brasil e no mundo. O Firebird nasceu com base no Interbase 6 mediante a uma abertura no código fonte deste último proporcinado pelo seu fabricante, ou seja, a própria Borland, que numa estratégia de mercado para difundir o Interbase, abriu seu código fonte e pouco depois, anunciou a versão 6.5 do Interbase, a qual não teria continuidade como "Open-Source". Mediante isso, uma comunidade liderada por Ann Harrison (considerada a "mãe" do Interbase) foi formada e hoje já temos a versão 1.5 do Firebird e milhares de usuários espalhados pelo mundo.

Uma das preocupações da Borland até o momento, têm sido compatibilizar seus componentes de acesso a banco de dados, mas especificamente ao Interbase com seu próprio produto, ou seja, atualmente o Interbase comercializado pela Borland está na versão 7.x e o Delphi 8 .NET naturalmente acessa este banco de dados como não poderia ser diferente, utilizando-se de um manage provider específico, o qual infelizmente não é 100% compatível com o Firebird, devido este último ter sofrido diversas implementações e correções de BUGs e dia-a-dia tomando rumos diferentes ao Interbase 7.x.

Isso contudo, não é problema, pois, mesmo antes da concepção do Delphi 8 .NET, um projeto "Open-Source" chamado *Firebird .NET Provider Data Provider* já estava em andamento, inclusive com uma release final disponível gratuítamente para a comunidade de usuários do Firebird, onde, este surgiu pela necessidade de acesso ao Firebird através de ferramentas para desenvolvimento .NET, como exemplo o Microsoft Visual Studio .NET. Vale ressaltar, que no Delphi 8 .NET a Borland implementou uma camada de acesso chamada BDP (Borland Data Provider) a qual é baseada na tecnologia ADO.NET e oferece ao programador Delphi maior produtividade devido sua integração com os componentes "DB Web", através dos quais podemos simplesmente ir relacionando componentes da mesma forma que estávamos habituados fazer nas versões anteriores do Delphi.

Bem, neste artigo iremos apresentar estas duas alternativas de conexão e demonstrar como utilizá-las, deixando a você a escolha de qual melhor irá atender suas necessidades.

Criando o Banco de Dados

Antes de demonstrarmos a conexão, necessitamos ter um banco de dados. Assim sendo, abra a ferramenta de administração do Firebird que esteja habituado a trabalhar, crie um novo banco, como sugestão nomeie "CADFB15.FDB". A seguir, crie uma tabela chamada CLIENTES, a qual irá conter a estrutura apresentada na *listagem 1*.

```
CREATE TABLE CLIENTES (
    ID INTEGER NOT NULL,
    NOME VARCHAR (80),
    ENDERECO VARCHAR (80),
    BAIRRO VARCHAR (30),
    CIDADE VARCHAR (30),
    UF CHAR (2),
    CEP CHAR (10),
    DOC1 VARCHAR (18),
    DOC2 VARCHAR (18),
    TELEFONE VARCHAR (14));
/* Chave Primária */
ALTER TABLE CLIENTES ADD CONSTRAINT PK_CLIENTES PRIMARY KEY (ID);
```

Listagem 1 – Criando a tabela

Para o campo **ID** vamos criar um *generator* e uma *trigger* para que o mesmo seja incrementado automaticamente, veja a *listagem 2.*

```
/* Generator */
CREATE GENERATOR GEN_CLIENTES_ID ;

/* Trigger */
CREATE TRIGGER TRG_CLIENTES_BI0 FOR CLIENTES ACTIVE
BEFORE INSERT POSITION 0
AS
begin
  NEW.ID = GEN_ID(GEN_CLIENTES_ID, 1);
end
```

Listagem 2 – Generator e Trigger

Agora vamos ao que realmente interessa, a conexão via BDP!

Acessando o Firebird via BDP

Junto ao Delphi 8 .NET a Borland disponibilizou drivers BDP para acesso a vários bancos de dados, sendo: DB2, Interbase,

MSAccess, MSSQL e Oracle. Bem, você pode estar pensando, posso acessar o Firebird utilizando o driver para Interbase assim como estou habituado a fazer no dbExpress, onde existe compatibilidade entre ambos, certo? Errado! Devido a várias implementações e correções efetuadas no Firebird esta compatibilidade não é 100% e o driver que acompanha o Delphi 8 .NET até consegue efetuar a conexão a um banco Firebird, contudo não consegue extrair informações de tabelas, índices, etc. Mediante isso, o mesmo autor do projeto *Firebird .NET Provider Data Provider,* iniciou a implementação de um driver BDP específico para o Firebird. Até o momento da conclusão deste artigo, a versão disponível é a versão Alpha 1 para o .NET Framework 1.1. Apesar de ser uma versão Alpha, executamos vários testes e os resultados forem surpreendentes, acredito que em breve teremos uma versão final. Bem, o primeiro passo é efetuar o download do "BDP For Firebird" no seguinte endereço:

http://www.ibphoenix.com/main.nfs?a=ibphoenix&s=1089111581:165&page=ibp_download_dotnet

A instalação é bastante simples, basta ir clicando em "Próximo" até o final. Bem, este é o primeiro passo da instalação. Antes de abrir o Delphi 8 .NET, iremos necessitar efetuar entradas nos arquivos BdpDataSources.xml e BdpConnections.xml, ambos disponibilizados na pasta ...\Borland\BDS\2.0\Bin. Abra estes arquivos através do *bloco de notas* por exemplo, e adicione as seguintes entradas ao final dos mesmos:

*** BdpDataSources.xml:**

```
<provider name="Firebird"
connectionStringType="FirebirdSql.Data.Bdp.FbConnectionString,
FirebirdSql.Data.Bdp, Version=1.0.0.0, Culture=neutral, PublicKeyToken=c7d0a028dd9e545b">
        <objectTypes>
        <objectType>Tables</objectType>
        <objectType>Procedures</objectType>
        <objectType>Views</objectType>
        </objectTypes>
</provider>
```

*** BdpConnections.xml:**

```
<BdpConnectionString xsi:type="FbConnectionString">
        <Name>FbConn1</Name>
        <Database>localhost/3050:employee.fdb</Database>
        <UserName>sysdba</UserName>
        <Password>masterkey</Password>
        <Assembly>FirebirdSql.Data.Bdp,Version=1.0.0.0,Culture=neutral,PublicKeyToken=c7d0a028dd9e545b</Assembly>
```

```
</BdpConnectionString>
```

Prosseguindo, acesse o "Data Explorer" e observe que agora temos um ítem "Firebird" o qual representa nossa "ponte" ao Firebird, veja a *figura 1.*

Figura 1 – Data Explorer

Para testar, vamos criar um nova aplicação ASP.NET e para isso, acesse o menu File | New | Asp.Net Web Application, de um nome que achar mais conveniente ou simplesmente clique em OK para confirmar a sugestão do Delphi. Estando no "Data Explorer", clique da direito sobre "Firebird" e selecione "Add New Connection", com isso será apresentada uma tela igual a *figura 2*.

Figura 2 – Adicionando nova conexão

Informe um nome para a conexão e clique OK. Feito isso, será adiciona a nova conexão, clique da direita sobre a mesma e selecione "Modify Connection", com isso será apresentado o "Connections Editor", conforme demonstra a *figura 3.*

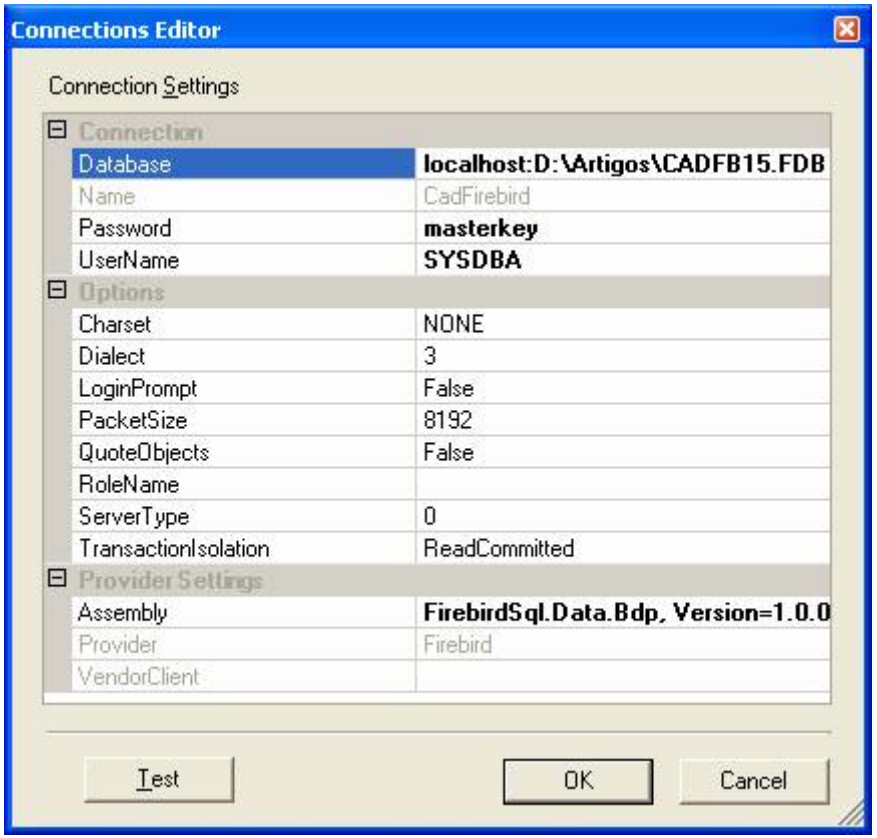

Figura 3 – Connections Editor.

Como pode observar, não temos nada de estranho aqui e sim muito pelo contrário. No "Connections Editor" iremos informar o path do banco de dados, usuário, senha, ou seja, tudo que é necessário para a conexão com nosso banco de dados. Após configurar os parâmetros, clique no botão "Test" para certificar-se de que tudo está OK e caso positivo, clique em OK. Voltando ao "Data Explorer", expanda nossa conexão clicando no (+) e poderá observar os ítens: Tables, Views e Procedures. Expanda o ítem Tables clicando no (+) irá visualizar a tabela CLIENTES. Clique sobre a tabela e arraste a mesma sobre seu WebForm. Com isso, automaticamente serão adicionados um componente BdpConnection e um BdpDataAdapter, nomeie como cnxClientes e adpClientes, respectivamente. Feito isso, vamos efetuar as configurações necessárias no adpClientes e gerar um objeto DataSet... Clique da direita sobre o adpClientes e selecione "Configure Data

Adapter", veja a *figura 4.*

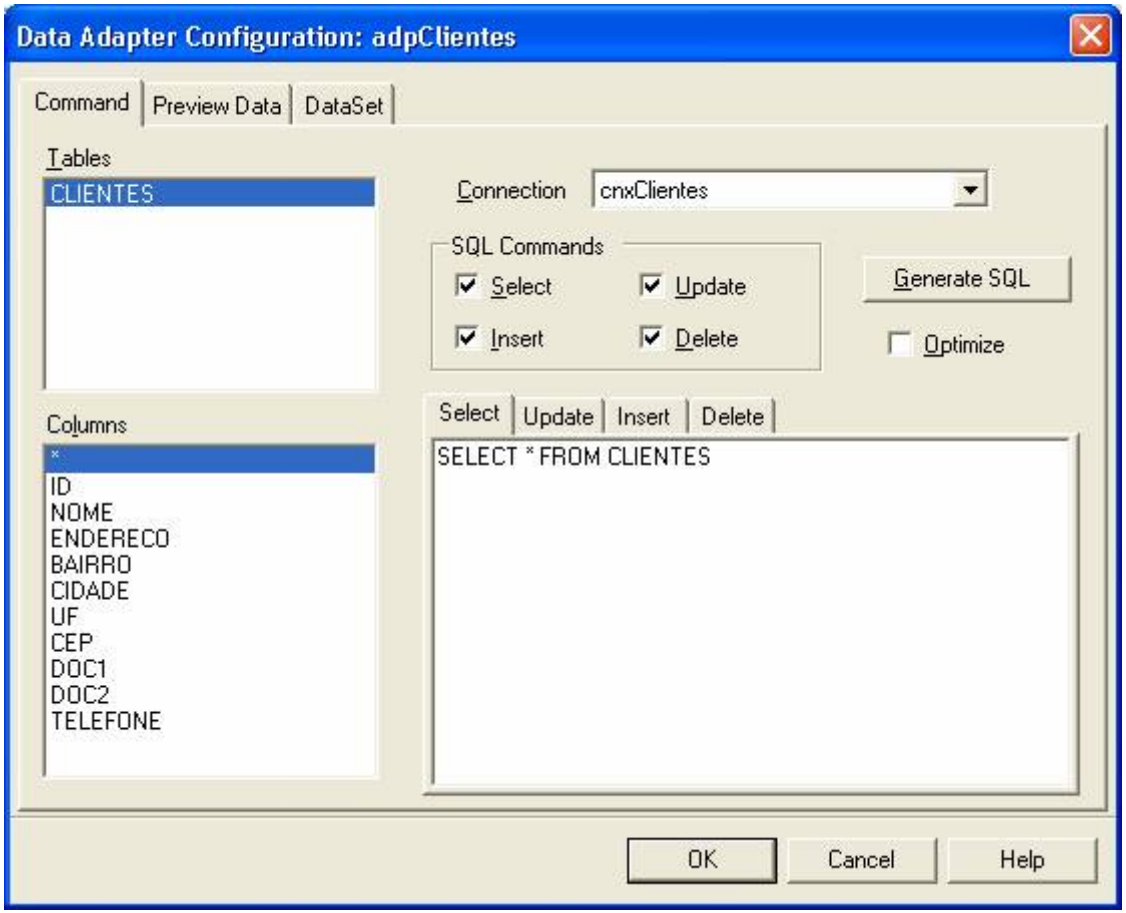

Figura 4 – Configuração do Data Adapter.

Vá até a aba "DataSet", clique na opção "New DataSet" e digite "dsClientes" e para finalizar clique OK, com isso será adicionado automaticamente um componente DataSet e para concluir a configurações dos componentes responsáveis ao acesso e manutenção no banco de dados, adicione um componente DBWebDataSource, disponibilizado na "Tool Palette" no grupo "DB Web", nomeio como srcClientes e configure sua propriedade *DataSource* para **dsClientes** (que é o nosso componente DataSet gerado via adpClientes). Agora, vá ao evento

OnApplyChangesRequest do srcClientes e adicione a seguinte instrução apresentada na *listagem 3.*

```
procedure TWebForm_CadCli.srcClientes_OnApplyChangesRequest(sender: System.Object;
  e: Borland.Data.Web.WebControlEventArgs);
begin
  adpClientes.AutoUpdate;
end;
```

Listagem 3 – Código no evento OnApplyChangesRequest.

Até aqui, implementamos o "motor" de nosso simples cadastro, o qual irá permitir adicionar, alterar, excluir, enfim, fazer toda a manutenção necessária em nossa tabela CLIENTES. Vamos agora implementar a interface onde o usuário irá interagir com nosso aplicativo. Comece por adicionar um componente DBWebNavigator disponibilizado na "Tool Palette" no grupo "DB Web" o qual iremos nomear navClientes, configure a propriedade *ButtonType* para *ButtonIcons,* a propriedade *DBDataSource* para *srcClientes* e a propriedade *TableName* igual a *CLIENTES*. Após isso, adicione componentes "Label" (disponibilizado na "Tool Palette" no grupo "Web Controls") e componentes "DBWebTextBox" (disponibilizado na "Tool Palette" no grupo "DB Web"), feito isso, iremos fazer a ligação destes DBWebTextBox com as colunas da tabela CLIENTES onde para isso iremos configurar a propriedade *DBDataSource* para *srcClientes* e a propriedade *TableName* igual a *CLIENTES* e a propriedade *ColumnName* para cada coluna da tabela. Na *figura 5* apresentamos uma sugestão para o layout.

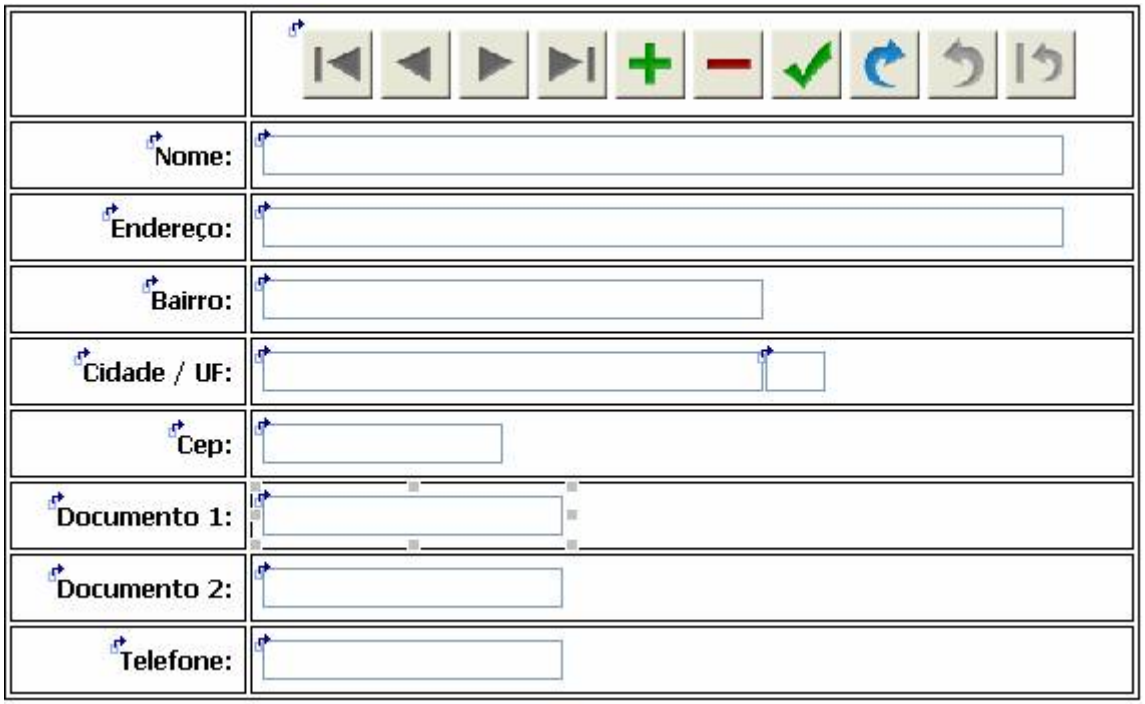

Figura 4 – Layout sugerido

Se todas as ligações estiverem corretas, nosso exemplo estará pronto e o próximo passo é só efetuar o teste, para isso, acesse o menu "Run" e escolha "Run Without Debugging", poderia simplesmente teclar <F9>, contudo, a execução "sem debug" é bem mais leve, use <F9> somente quando necessitar utilizar o debug. A *figura 5* demonstra nosso aplicativo em execução.

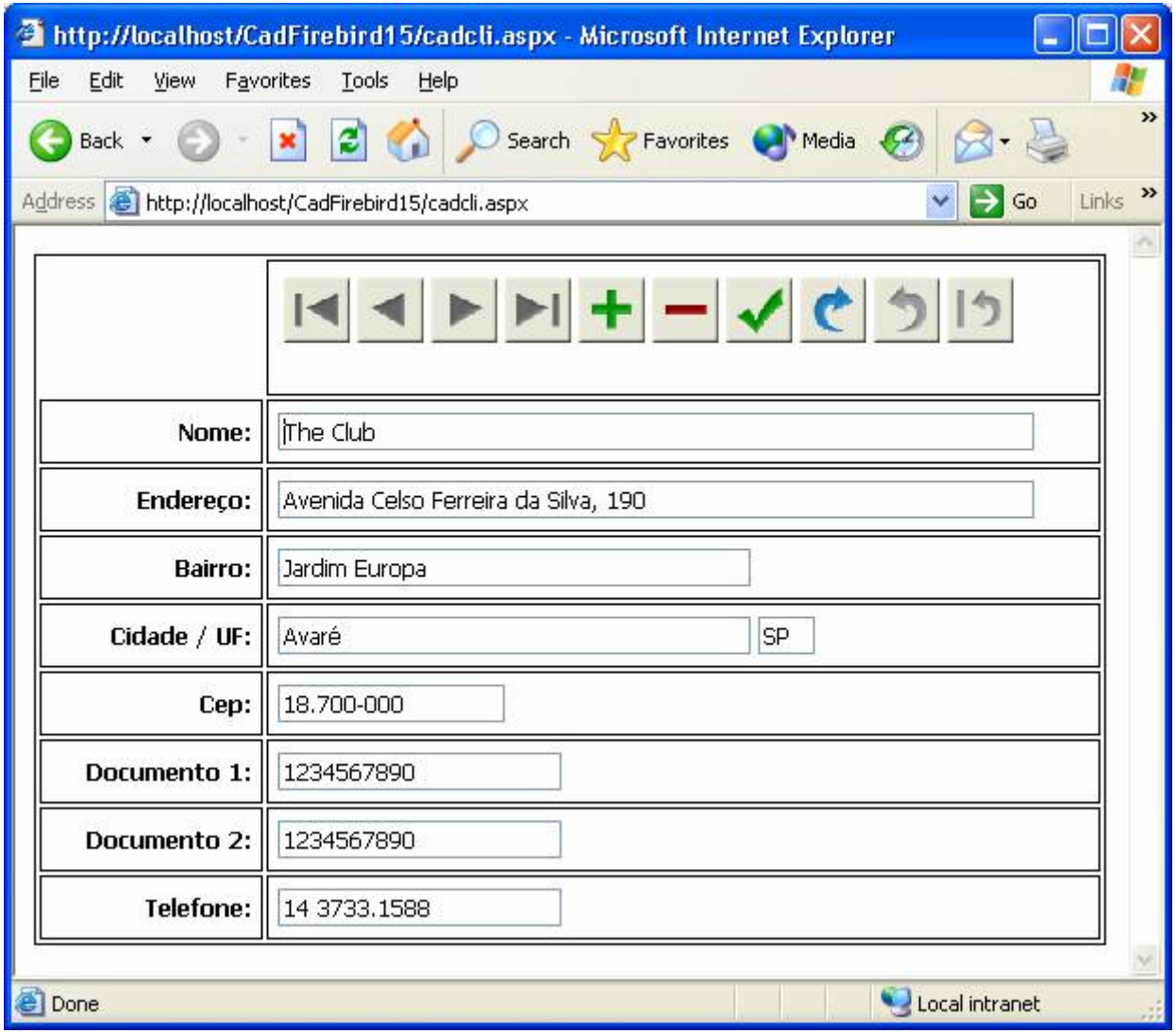

Figura 5 – Exemplo em execução

Conclusão

Nesta primeira parte demonstramos como efetuar a conexão com o Firebird utilizando a tecnologia BDP graças ao driver disponibilizado pelo Carlos Guzmán Álvarez, autor e mantenedor do Borland Data Provider For Firebird, bem como do Firebird .NET Provider, o qual iremos abordar na próxima edição da The Club Megazine, contudo, gostaria de ressaltar que trabalhar com BDP é muito mais simples e produtivo que os demais manage providers existentes para a plataforma .NET.

Autor: Alessandro Ferreira alessandro@theclub.com.br

Download

Clique aqui para baixar o exemplo apresentado neste artigo.

Avalie esse artigo : Excelente Envia

Voltar