

## ADO Express (ADOX)

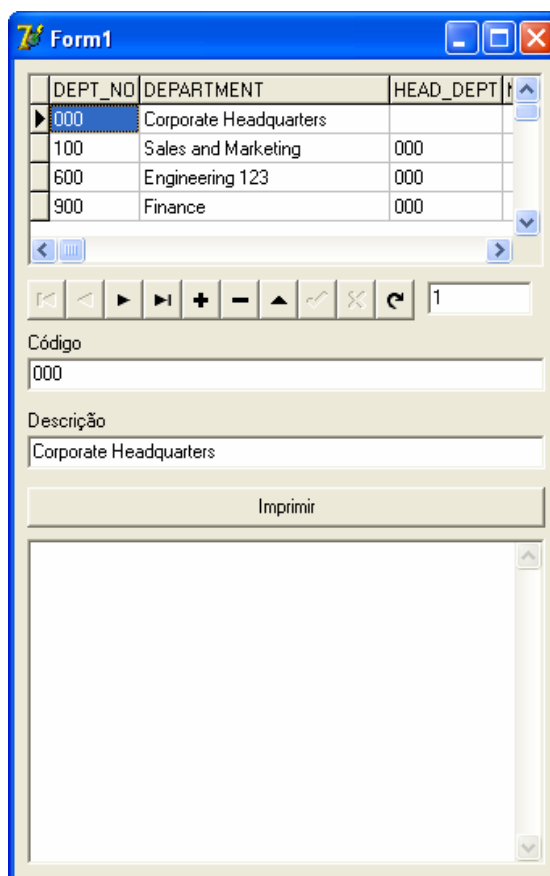
### Introdução

A API ADO Express utiliza a tecnologia OLEDB da MICROSOFT para acessar dados das mais diversas bases de dados. É necessário ter instalado na máquina o pacote MDAC ( Microsoft Data Access Components) atualmente na versão 2.8, que já vem instalado nos sistemas operacionais Windows mais recentes como o Windows 2000 e Windows XP. Além disso é necessário instalar o DRIVER de acesso ao banco de dados que se deseja utilizar, no caso do Firebird o IBOLE.

O componente TADOConnection é responsável pela conexão com o Banco de Dados, sendo necessário apenas 1 por programa. O componente TADOQuery faz a consulta SQL ao Banco de Dados e conecta o resultado desta consulta aos componentes de tela através de um componente DataSource.

TADOConnection <= TADOQuery <= TDataSource <= TDBGrid, TDBNavigator, TDBEdit

### Programa



The screenshot shows a Delphi application window titled "Form1" with a blue title bar. The main content area contains a data grid with the following data:

DEPT_NO	DEPARTMENT	HEAD_DEPT
000	Corporate Headquarters	
100	Sales and Marketing	000
600	Engineering 123	000
900	Finance	000

Below the grid is a set of navigation buttons (back, forward, home, end, refresh, etc.) and a page number field showing "1". Underneath are two text boxes: "Código" containing "000" and "Descrição" containing "Corporate Headquarters". At the bottom, there is an "Imprimir" button and a large empty text area.

## Componentes

TADOConnection: faz a conexão com o Banco de Dados

**Connection:** clicar 2 vezes sobre o componente e definir os seguintes parâmetros:

Driver: Zstyle IBOLE Driver

DataSource: localhost:C:\Arquivos de Programas\Firebird\examples\EMPLOYEE.fdb

“localhost” pode ser substituído por “127.0.0.1” ou pelo IP da máquina do servidor de dados

User Name: SYSDBA

Password: masterkey

**LoginPrompt:** False

**Connected:** True

TADOQuery: faz a consulta SQL

**Connection:** ADOConnection1

**SQL:** SELECT \* FROM DEPARTMENT

**Active:** True, abre a consulta

TDataSource: faz a conexão da consulta com os componentes de tela

**DataSet:** ADOQuery1

TDBGrid: grid de acesso a dados

**DataSource:** DataSource1

TDBNavigator: controle de navegação de dados

**DataSource:** DataSource1

**Hints:** lista de “hints” em Inglês, que pode ser traduzida

**ShowHint:** True, mostra os “hints” dos botões

TDBEdit: componente de edição de dados ( ao estilo do TEdit )

**DataField:** DEPT\_NO e DEPARTMENT

**DataSource:** DataSource1

## Acesso a Campos de Consultas ( Queries ) via Código

O acesso a campos de uma consulta via código é feita através dos métodos “Fields”, que acessa o Nésimo campo da consulta ( começando de 0 ) ou “FieldByName” que acessa um campo pelo nome. O exemplo considera que a consulta retorna 2 campos: DEPT\_NO e DEPARTMENT.

*ADOQuery1.FieldByName('DEPT\_NO').AsString*, volta o campo “DEPT\_NO” como string

*ADOQuery1.Fields[1].AsString*, volta o campo “DEPARTMENT” como string

O objeto "TField" define um campo de uma consulta, sendo um tipo abstrato que é instanciado para um dos tipos abaixo, sendo os mais comuns os tipos TStringField, TIntegerField, TFloatField e TNumericField:

TADTField  
TAggregateField  
TArrayField  
TAutoIncField  
TBCDField  
TBinaryField  
TBlobField  
TBooleanField  
TBytesField  
TCurrencyField  
TDataSetField  
TDateField  
TDateTimeField  
TFloatField  
TFMTBCDField  
TGraphicField  
TGuidField  
TIDispatchField  
TIntegerField  
TInterfaceField  
TLargeintField  
TMemoField  
TReferenceField  
TSmallIntField  
TSQLTimeStampField  
TStringField  
TVarBytesField  
TVariantField  
TWideStringField  
TWordField

Existem várias formas de retornar o campo, inclusive com conversão automática entre tipos, podendo por exemplo, um campo "TIntegerField", que contém um valor inteiro, usar o método AsString para converter o inteiro em string.

AsCurrency: Currency  
AsDate: TDateTime  
AsDouble: Double  
AsInteger: Integer  
AsString: string

## Navegação de Consultas ( Queries ) via Código

As consultas podem ser “navegadas”, ou seja, podemos passar de registro a registro como se fosse uma planilha de cálculo onde as linhas representam os registros e as colunas os campos. Os comandos de navegação podem ser:

First: ir para o primeiro registro

Prior: ir para o registro anterior

Next: ir para o registro posterior

Last: ir para o último registro

RecNo: retorno o número do registro atual ( começando de 0 )

Pode-se, por exemplo, criar um método para tratar o evento “DataChange” do TDataSource, que ocorre a cada vez que movemos o registro da consulta:

```
procedure TForm1.DataSource1DataChange(Sender: TObject; Field: TField);  
begin  
    Edit1.Text := IntToStr(ADOQuery1.RecNo);  
end;
```

## Operações de Manipulação de Banco de Dados

As operações de manipulação de bancos de dados podem ser executadas através de métodos da classe TADOQuery:

TADOQuery.Insert, cria um novo registro para inclusão

TADOQuery.Edit, abre um registro existente para alteração

TADOQuery.Delete, excluir um registro

A operação de Exclusão (Delete) não necessita de confirmação, sendo executada diretamente no banco de dados, mas as operação de Inclusão (Insert) e Alteração (Edit) necessitam ser confirmadas ou canceladas com:

TADOQuery.Post, grava as alterações

TADOQuery.Cancel, ignora as alterações, não criando, no caso da Inclusão, ou não alterando no caso da Alteração

Novamente o evento “DataChange” do TDataSource pode ser usado para monitorar alterações do estado de um objeto da classe TADOQuery, que é representado pela propriedade “State” que pode assumir valores como:

dsBrowse: indica que o objeto está apenas navegando nos registros

dsInsert: indica que um registro está sendo incluído

dsEdit: indica que um registro está sendo alterado