

AJAX

Usando com o Delphi a biblioteca MagicAjax



FABRÍCIO DESBESSEL

(fabricio.desbessel@terra.com.br) é professor de Linguagem de Programação do Curso Técnico em Informática do Colégio Frederico Jorge Logemann de Horizontina/RS e da FAHOR Faculdade Horizontina. Delphiano de coração está

sempre disposto a provar que com o Delphi sempre teremos a melhor solução. Site www.fabricio.pro.br.

Existem inúmeras formas de utilizar a tecnologia AJAX em aplicações .NET, mas nenhuma apresenta tanta facilidade como a utilização do framework *MagicAjax.Net*. Esse framework é gratuito, está na versão beta e pode ser baixado no endereço oficial do projeto: www.magicajax.net.

Com a sua utilização, não é necessário ter o conhecimento de Java Script para criar as funções que enviam e buscam (*callback*) as informações sem dar um *refresh* total na página, diminuindo o tráfego de informações entre o servidor e os clientes, consumindo menos tempo.

Além disso, ele automaticamente apresenta a mensagem de *Loading* no canto superior direito, da mesma forma que acontece no *Gmail*. Neste artigo não vou entrar na parte teórica e introdutória sobre o AJAX, pois isso já foi tratado em outro artigo da revista Clube Delphi na edição 70. Para uma rápida introdução sobre a tecnologia, veja o box “O que é AJAX?”.

O que é AJAX?

Quando navegamos em páginas Web, sempre que é necessário efetuar uma comunicação com o servidor (clique de um botão, por exemplo), há um intenso tráfego de dados e toda a página é sempre recarregada (*refresh* total). Com tecnologias como o AJAX (Asynchronous JavaScript and XML), podemos limitar os postbacks ao servidor, evitando *refresh*s totais da página. Podemos chamar métodos de forma assíncrona e então ajustar pequenas porções da tela, diminuindo o tráfego de dados na rede. Com isso, aplicações Web se tornam semelhantes a aplicações Desktop. Trabalhar com AJAX “puro” requer vasto conhecimento de Java Script e exige codificação exaustiva. Dessa forma, várias tecnologias e ferramentas definem frameworks para tornar o trabalho sobre o AJAX mais produtivo, em várias linguagens e plataformas diferentes (Java, .NET etc.). Um desses frameworks é o *MagicAjax*.

Criando uma consulta a dados com Firebird

No Delphi 2005 ou 2006, crie um novo projeto do tipo *ASP.NET Web Application - Delphi for .NET*. Preencha o *Name* com "Magico" e clique em OK. Vamos criar uma consulta de dados nas tabelas *Department* e *Employee* do banco *Employee.fdb*, através do *Firebird Data Provider* (veja box). Adicione um *FbConnection*, clique na propriedade *ConnectionString* e configure suas propriedades conforme a **Figura 1**.

Teste a conexão clicando no botão *Test* e depois clique no *Accept*. Agora adicione um *FbCommand*, configurando sua propriedade *Connection* para o *fbConnection1* e a propriedade *CommandText* conforme a **Listagem 1**.

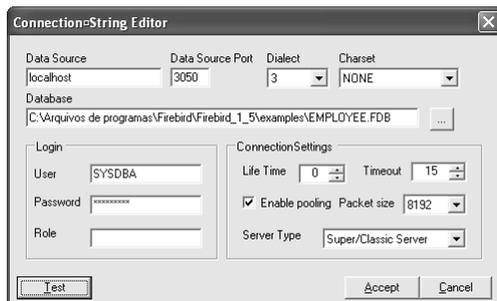


Figura 1. Configurações da conexão ao Firebird

Firebird Data Provider

Para acessar o Firebird a partir de aplicações ASP.NET, a melhor opção é usar o provider do banco para o ADO.NET, chamado Firebird Data Provider. Para instalá-lo, acesse o endereço www.firebirdsql.com, clique no link *Download* a seguir em *Firebird .NET Data Provider*. Baixe e instale a última versão do *Data Provider for .NET Framework 1.1*, bastando seguir os passos no assistente que será iniciado. Neste artigo usaremos versão 1.7 do *Provider*, versão mais recente disponível até o fechamento desta edição. Após a instalação, no Delphi 2005/2006 clique no menu *Component | Installed .NET Components*. Digite "Firebird Data Provider" na opção *Category*, clique no botão *Select an Assembly* e escolha o arquivo *FirebirdSql.Data.Firebird.dll*, localizado no diretório de instalação do *Provider*, por padrão em *C:\Arquivos de programas\FirebirdNETProvider(versão o)*. Clique em *Ok* e observe que os novos componentes para acesso ao Firebird estão agora disponíveis no IDE. Maiores informações sobre o funcionamento e utilização do *Firebird Data Provider* podem ser encontradas na edição 66 da *ClubeDelphi*, ou ainda no curso de Delphi e ASP.NET da *DevMedia* (<http://www.devmedia.com.br/curso/ecommerce2005>).

Borland® E-Commerce

www.borlandshop.com.br

O seu canal direto de compras para produtos Borland

Mais Fácil

Mais Seguro

Mais Econômico





Listagem 1. Instrução SQL do fbCommand

```
select
  E.EMP_NO,
  E.FIRST_NAME,
  E.LAST_NAME,
  D.DEPARTMENT
from
  EMPLOYEE E
inner join
  DEPARTMENT D on E.DEPT_NO = D.DEPT_NO
where upper(FIRST_NAME) like @FIRST_NAME
order by
  E.FIRST_NAME
```

Abra o editor da propriedade *Parameters* do *FbCommand* e adicione um novo parâmetro, configurando seu *ParameterName* para “@First_Name” e *SourceColumn* para “First_Name”. Verifique se o *FbDbType* está como *VarChar*.

Adicione um *TextBox* e um *Button* ao formulário. Na propriedade *Text* do *Button* digite “Pesquisar”. Adicione também um *DataGrid*. Clique duas vezes sobre o botão e codifique conforme a **Listagem 2**.

Listagem 2. Código do botão Pesquisar

```
FbConnection1.Open;
try
  FbCommand1.Parameters['@FIRST_NAME'].Value :=
    TextBox1.Text.ToUpper + '%';
  DataGrid1.DataSource := FbCommand1.ExecuteReader;
  DataGrid1.DataBind;
finally
  FbConnection1.Close;
end;
```

Você pode compilar a aplicação e testar informando alguma letra ou nome. Para trazer todos os registros deixe o *TextBox* em branco em clique em *Pesquisar*. Note que a cada pesquisa a página será toda atualizada (há o refresh total da página). Até agora, simplesmente montamos uma pesquisa com Firebird e ainda não estamos utilizando a tecnologia AJAX. Vamos ao próximo passo.

Utilizando o MagicAjax

Em primeiro lugar precisamos adicionar uma referência para a DLL do *MagicAjax* para que possamos usar a tecnologia. No *Project Manager* selecione o item *References* e clique com o botão direito do mouse para abrir o menu de contexto e escolha *Add Reference*. Na tela de *Add Reference*, com a tab *.NET Assemblies* selecionada, clique no botão *Browse* e encontre o arquivo *MagicAjax.dll* a partir do diretório onde você o instalou. Clique em OK para fechar o editor.

Também no *Project Manager*, clique duas vezes no ar-

quivo *web.config* pois precisaremos editá-lo, informando à aplicação para tratar as requisições de forma diferenciada, através do *MagicAjax*. Procure pela sessão `<httpModules>` e adicione o seguinte código:

```
<add name="MagicAjax" type="MagicAjax.MagicAjaxModule, MagicAjax"/>
```

Abra o *WebForm1* e exiba seu código ASPX, pois teremos que registrar um *NameSpace* e criar um objeto *panel* do AJAX que dirá para a aplicação que todo o processamento programado nos componentes dentro desse painel será processado via AJAX, sem a necessidade de dar um *refresh* na página inteira. Para registrar informamos:

```
<% Register TagPrefix="ajax" Namespace="MagicAjax.UI.Controls"
  Assembly="MagicAjax" %>
```

Para criar um painel, adicione:

```
<ajax:AjaxPanel id="AjaxPanel1" runat="server">
</ajax:AjaxPanel>
```

Veja o código completo do arquivo ASPX na **Listagem 3**.

Listagem 3. Código completo do ASPX

```
<% Page language="c#" Debug="true"
  CodeBehind="WebForm1.pas" AutoEventWireup="false"
  Inherits="WebForm1.TWebForm1" %>
<% Register TagPrefix="ajax" Namespace="MagicAjax.UI.Controls"
  Assembly="MagicAjax" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title></title>
  </head>
  <body>
    <form runat="server">
      <ajax:AjaxPanel id="AjaxPanel1" runat="server">
        <p><font face="Verdana" size="2">
          <strong>Pesquisa com Ajax</strong>
        </font>
        </p>
        <p>
          <hr width="100%" size="1">
          <ASP:TextBox id="TextBox1" runat="server">
          </ASP:TextBox>

          <ASP:Button id="Button1" runat="server" text="Pesquisar">
          </ASP:Button></p>
        <p></p>
        <p>
          <ASP:DataGrid id="DataGrid1" runat="server">
          </ASP:DataGrid>
        </p>
      </ajax:AjaxPanel>
    </form>
  </body>
</html>
```

Feito isso, você pode compilar a aplicação e fazer buscas no banco de dados. Note que ao clicar no botão, aparecerá no canto superior direito a informação *Loading* (**Figura 2**). Isso indica que a aplicação está usando a tecnologia AJAX para buscar o resultado da pesquisa. Outro fator importante é que não precisamos colocar nenhum código Java Script para utilizar a tecnologia, o que difere de outras bibliotecas que necessitam de uma codificação extra.

Nota: Dependendo da quantidade de registros, a mensagem de *Loading* pode aparecer rapidamente de forma quase imperceptível.

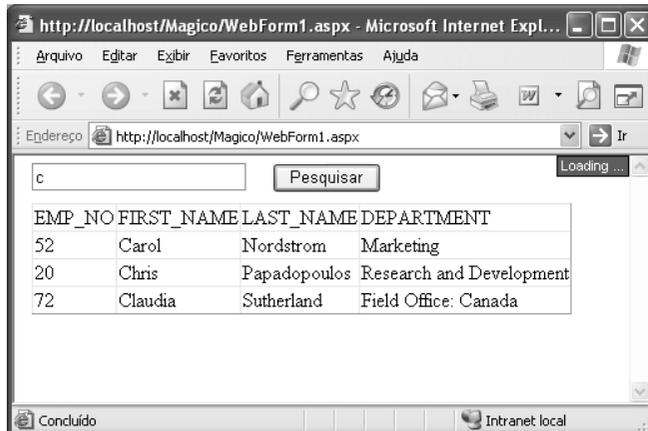


Figura 2. Aplicação com AJAX em funcionamento

Listando opções

Agora vamos criar uma nova aplicação com listas de seleções. A idéia, é primeiramente escolher um país, depois escolher um cliente daquele país para ver seus dados. Crie uma nova aplicação e dê o nome de "Clientes". Para demonstrar que a página não será totalmente recarregada, coloque um *HTML Image* da paleta *HTML Elements*. Clique com o botão direito do mouse sobre o componente e escolha *Properties*. Para *Image Source* informe: "http://www.devmedia.com.br/Imagens/logo.gif".

Logo abaixo escreva o texto "Pesquisa de Consumidores". Adicione um *HTML Horizontal Rule*, também da paleta *HTML Elements*. Abaixo, escreva o texto "País:", pressione Shift+ENTER e coloque um *DropDownList*. Pressione ENTER e escreva o texto "Consumidor:". Pressione Shift+ENTER e coloque outro *DropDownList*. Por último, adicione um *DataGrid* e configure sua aparência, clicando em *Auto Format*, ao final do *Object Inspector*. Veja como deve ficar a interface final na **Figura 3**.

Adicione um *FbConnection*, clique na propriedade *ConnectionString* e configure suas propriedade conforme a **Figura 1**. Coloque três componentes do tipo *FbCommand*, apontando suas propriedades *Connection* para o *fbConnection1* e a propriedade *CommandText* conforme a **Listagem 4**.

Listagem 4. Código SQL dos componentes Command

Instrução SQL do FbCommand1

```
select COUNTRY
from COUNTRY
order by COUNTRY
```

Instrução SQL do FbCommand2

```
select CUST_NO, CUSTOMER
from CUSTOMER
where COUNTRY = @COUNTRY
order by CUSTOMER
```

Instrução SQL do FbCommand3

```
select CUST_NO, CUSTOMER, CONTACT_FIRST, CITY
from CUSTOMER
where CUST_NO=@CUST_NO
```

Note que no *FbCommand2* e no *FbCommand3* utilizaremos parâmetros nas instruções SQL. Então precisamos

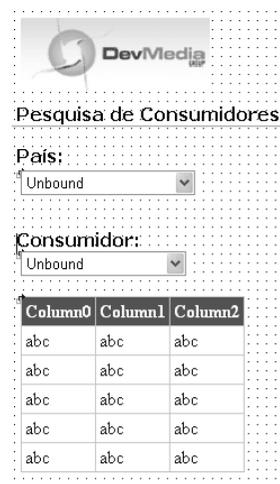


Figura 3. Aparência final da interface

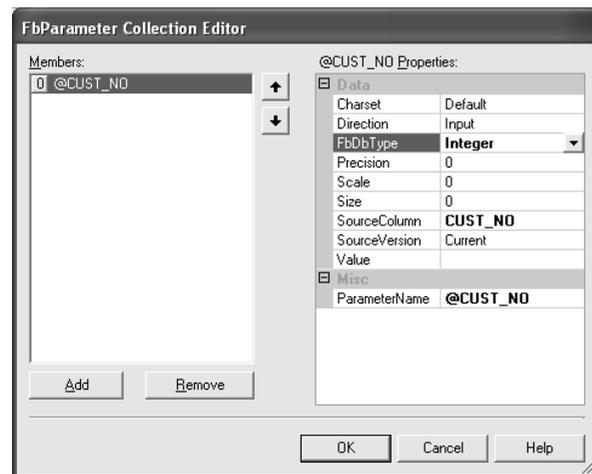


Figura 4. Configuração da propriedade Parameters do FbCommand3

adicionar esses parâmetros na propriedade *Parameters*. Selecione o *FbCommand2* e clique na propriedade *Parameters*. No editor de parâmetros que será aberto, clique no botão *Add*, para a propriedade *ParameterName* informe "@COUNTRY" e em *SourceColumn* informe "COUNTRY".

Repita esses passos no *FbCommand3* informando *ParameterName* como "@CUST_NO" e *SourceColumn* com "CUST_NO". Como nesse caso o parâmetro será inteiro é necessário alterar a propriedade *FbDbType* para *Integer*, conforme mostra a **Figura 4**. Vamos ao código. Para o evento *Load* do *WebForm1* digite o código contido na **Listagem 5**.

Listagem 5. Código do evento Load do WebForm

```
if not Page.IsPostBack then
begin
FbConnection1.Open;
try
DropDownList1.DataTextField := 'COUNTRY';
DropDownList1.DataValueField := 'COUNTRY';
DropDownList1.DataSource :=
FbCommand1.ExecuteReader;
DropDownList1.DataBind;
finally
FbConnection1.Close;
end;
end;
```

Clique duas vezes sobre o *DropDownList1* e codifique-o conforme a **Listagem 6**.

Listagem 6. Código do evento *SelectedIndexChanged* do *DropDownList1*

```
FbConnection1.Open;
try
    DropDownList2.DataTextField := 'CUSTOMER';
    DropDownList2.DataValueField := 'CUST_NO';
    FbCommand2.Parameters['@COUNTRY'].Value :=
        DropDownList1.SelectedValue;
    DropDownList2.DataSource :=
        FbCommand2.ExecuteReader;
    DropDownList2.DataBind;
finally
    FbConnection1.Close;
end;
```

Agora clique duas vezes no *DropDownList2* e coloque o código conforme a **Listagem 7**.

Listagem 7. Código do evento *SelectedIndexChanged* do *DropDownList2*

```
FbConnection1.Open;
try
    FbCommand3.Parameters['@CUST_NO'].Value :=
        DropDownList2.SelectedValue;
    DataGrid1.DataSource := FbCommand3.ExecuteReader;
    DataGrid1.DataBind;
finally
    FbConnection1.Close;
end;
```

Para que o evento *SelectedIndexChanged* seja disparado pela aplicação, é necessário modificar a propriedade *AutoPostBack* do *DropDownList* para *True*. Faça isso nos dois componentes que foram adicionados em nossa aplicação.

Nesse momento você poderá testar a aplicação que já deve estar funcionando da forma normal, executando *Refresh* integral da página. Note que, ao selecionar algum país, o símbolo do IE (no canto superior direito) fica animado, informando que a página está sendo atualizada.

Agora é a hora de transformar a aplicação para usar o AJAX. No *Project Manager*, clique duas vezes no arquivo *web.config* para abri-lo. Procure pela sessão `<httpModules>` e adicione o código conforme o exemplo anterior (não esqueça de adicionar no *References* a DLL do AJAX).

Abra o *WebForm1* e exiba seu código ASPX para registrar a *NameSpace* do AJAX e criar um painel. Para registrar informe o seguinte código na segunda linha do arquivo:

```
<% Register TagPrefix="ajax"
    Namespace="MagicAjax.UI.Controls"
    Assembly="MagicAjax" %>
```

Para criar o painel, antes do primeiro *DropDownList*, digite o seguinte código:

```
<ajax:AjaxPanel id="AjaxPanel1" runat="server">
```

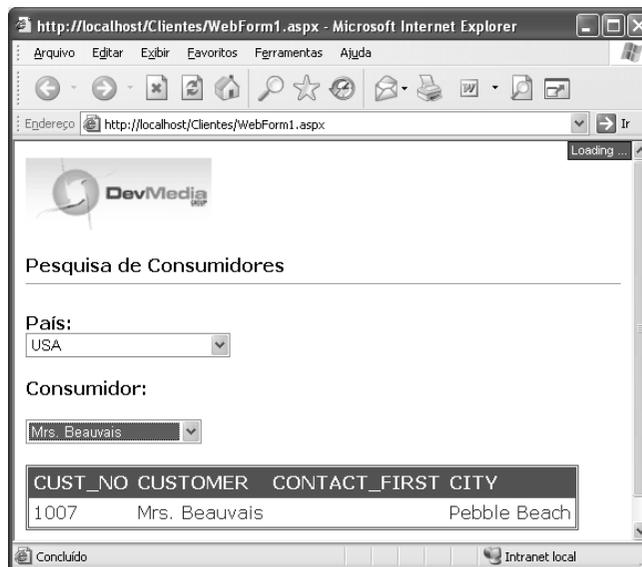


Figura 5. Aplicação com busca assíncrona dos dados, usando AJAX

Isso deve ficar antes da declaração do *DropDownList*:

```
<ASP:DropDownList id="DropDownList1" runat="server"
width="185px" autopostback="True"></ASP:DropDownList>
```

Para finalizar o painel, antes da linha: `</form>`, digite: `</ajax:AjaxPanel>` (assim fechamos o painel depois do *DataGrid*). Pronto, com isso você poderá testar a aplicação e ver que as informações da segunda lista de opções são buscadas através do AJAX, assim como a grade de informações do cliente/consumidor escolhido (**Figura 5**).

Isso pode ser confirmado, pois o símbolo do IE (no canto superior direito) não fica animado quando um país ou consumidor é selecionado.

Conclusão

O *MagicAjax* facilita em muito a criação de aplicações ASP.NET que usam a tecnologia AJAX. Essa facilidade irá difundir esse tipo de aplicação que economiza tráfego de informações entre clientes e servidores. Seja inovador, comece a utilizar essa tecnologia em suas aplicações. ■

Links

Treinamento on-line exclusivo em Delphi e ASP.NET da DevMedia
www.devmedia.com.br/curso/ecommerce2005/

