O Delphi e a Internet – Segundo artigo da série

Ao ler o artigo anterior você passou a ter uma boa idéia do que é programar em Delphi para a internet. Na verdade, falamos muito sobre o mecanismo de funcionamento da própria internet, bem como dos padrões CGI, ISAPI e NSAPI. Discutimos ainda as dificuldades que envolvem a programação para internet e sobre a necessidade de se possuir um serviço web em seu micro a fim de fazer os devidos testes durante a fase de desenvolvimento.

O que o Delphi nos possibilita a nível de internet/intranet é criar poderosos extensores web, similares aos próprios extensores ISAPI e NSAPI. E note que vem crescendo muito o desenvolvimento Delphi para intranet, tornando o domínio da programação Delphi/internet algo essencial para o programador moderno.

Mas como é efetivamente a programação Delphi/Internet?

CONSTRUINDO APLICAÇÕES PARA A INTERNET/INTRANET COM DELPHI

No artigo anterior, mencionamos que o mecanismo da internet baseia-se em requisições (request) e respostas (response). O cliente faz uma requisição e o servidor fornece uma resposta. Mas antes de dar uma resposta, o servidor processa alguma informação, o que provavelmente requer acesso a dados.

Assim, uma aplicação Delphi/Internet/Intranet precisa possuir meios para receber requisições e enviar respostas, além de alguma forma de acesso a dados. E é exatamente isto que o Delphi disponibiliza para você, programador.

Para criar um projeto novo com as características mencionadas acima, clique em "File/New". Selecione o ícone "Web Server Application" e pressione o botão "OK". As opções disponíveis são "ISAPI/NSAP Dynamic Link Library", "CGI stand-alone executable" e "Win-CGI stand-alone executable". Selecione a primeira opção, "ISAPI/NSAP Dynamic Link Library", e pressione o botão "OK". Um projeto novo é criado.

Este projeto consiste unicamente de um TWebModule, que descende do TDdataModule. Trata-se um projeto para a criação de DLL, o que significa que não pode ser executado normalmente. E deve ser assim mesmo, pois como dissemos no artigo anterior, isto permitirá que a aplicação seja carregada apenas uma vez no servidor web, mesmo que diversos clientes web estejam acessando a DLL em um mesmo momento. Esta aplicação pode entender o protocolo HTTP, tornando possível o recebimento de requisições e o envio de respostas.

Ao TWebModule podem ser adicionados apenas componentes não visuais, como se dá também com a sua classe pai, o TDataModule. Nele podem ser adicionados componentes para tratamento de dados, tais como TDataBase, TTable e TQuery, bem como componentes internet que produzam conteúdo HTML.

Mas o grande segredo do TWebModule é a propriedade *Actions*. Você pode criar vários *Actions* em sua aplicação, e cada um destes *Actions* possui um nome. Quando o cliente faz uma requisição para sua aplicação no servidor baseada naquele nome, um código ligado a ele é executado, e um resultado HTML é devolvido para o cliente em forma de resposta. Por exemplo, nesta aplicação que estamos desenvolvendo criarmos um *Action* de nome "teste", podemos adicionar código para este *Action.* Supondo que o nome da aplicação seja "aplicacao.dll" e esteja rodando no servidor web, o código do *Action* "teste" seria executado ao informarmos o seguinte URI no browser:

```
http://alfamicro.com.br/aplicacao.dll/teste
```

Note que o "/teste" no final está dizendo para "aplicacao.dll" executar o código do *Action* "teste".

Para criar os *Actions* em sua aplicação, selecione o TWebModule e no Object Inspector dê um duplo clique na propriedade *Action*. Isto fará aparecer a caixa de diálogo Actions. Adicione um novo *Action* pressionando o botão "Add". Note que isto cria um novo *Action* com o nome padrão. Selecione este Action e mude a propriedade "PathInfo" para "teste". Então selecione a guia de eventos no Oject Inspector e dê um duplo clique no envento "OnAction". Você verá algo assim:

```
procedure TWebModule1.WebModule1WebActionItem1Action(Sender: TObject;

Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);

begin

end;
```

Este *event handler* contém toda a informação referente à requisição que invocou este *Action,* bem como a informação para gerar a resposta. As informações da requisição do cliente estão no parâmetro "Request", que é do tipo TWebRequest. O parâmetro "Response" é do tipo TWebResposne, e será usado para retornar as informações necessárias ao cliente. Através deste *event handler* você pode escrever qualquer código necessário para responder a requisição, incluindo manipulação de arquivos, acesso a banco de dados, ou qualquer outra coisa necessária para montar um HTML de resposta para o cliente.

Esta página HTML deve ser montada na hora por sua aplicação. Assim, seria bom obter algum conhecimento sobre páginas HTML a fim de desenvolver aplicações Delphi/Internet/Intranet. Dentro de alguns dias, será iniciada uma série de artigos sobre internet, incluindo HTML e scripts. Será publicado em WWW.ALFAMICRO.COM.BR, em artigos, na seção internet.

Vamos adicionar código ao nosso exemplo para retornar uma página HTML simples. Usaremos um TStringList para armazenar a página HTML:

```
procedure TWebModule1.WebModule1WebActionItem1Action(Sender: TObject;

Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);

Var

            sPagina:TStringList;

begin

            sPagina:=TStringList.Create;

            Try

                        sPagina.Add('<HTML>');

                        sPagina.Add('<HEAD>');

                        sPagina.Add('<TITLE>Exemplo de uma aplicação
                        Delphi/Internet/Intranet</TITLE>');

                        sPagina.Add('</HEAD>');

                        sPagina.Add('<BODY>');

                        sPagina.Add('<B>Esta página foi criada por uma aplicação
                        Delphi<B>');

                        sPagina.Add('<HR>');

                        sPagina.Add('Viu como foi fácil?');

                        sPagina.Add('</BODY>');

                        sPagina.Add('</HTML>');

                        Response.Content:=sPagina.Text;

            Finally

                        sPagina.Free;

            End;

            Handled:=True;

end;
```

Compile a aplicação e copie a DLL resultante para o seu site, no servidor de testes. Chame o seu browser e forneça o endereço do seu site de teste, o nome da DLL gerada e adiciona "/teste", que é o "PathInfo" do *Action* a ser executado. O resultado em seu browse será algo assim:

Esta página foi criada por uma aplicação Delphi

Viu como foi fácil?

Agora, adicione um componente TTable ao TWebModule, e ligue-o ao alias DBDEMOS e a tabela Clients.dbf. Altere a propriedade "Name" deste TTable para tblClientes. Adicione o seguinte código antes da linha "*sPagina.Add('</BODY>');*":

```
sPagina.Add('<HR>');

tblClientes.Open;

While not tblClientes.EOF do

Begin

            sPagina.Add('<BR>'+tblClientes.FieldByName('Last_Name').AsString+','+tblC
            lientes.FieldByName('First_Name').AsString+'</BR>');

            tblClientes.Next

End;

tblClientes.Close;
```

Compile o projeto novamente e substitua a DLL anterior por esta nova. Lembre-se que para fazer isso deverá parar o serviço web, e reiniciá-lo após a cópia. O resultado obtido no browser deverá ser algo assim:

Esta página foi criada por uma aplicação Delphi

Viu como foi fácil?

Davis,Jennifer

Jones,Arthur

Parker,Debra

Sawyer,Dave

White,Cindy

Ou seja, além de retornar a informação anterior, a nossa aplicação no servidor retorna também dados de uma tabela.

Este projeto simples serviu para mostrar como programar para internet/intranet com o Delphi. Sem dúvida, é possível adicionar outros recursos às nossas aplicações. Contudo, creio que este artigo será de grande ajuda para você começar a fazer seus testes, e familiarizar-se com esta forma de programação, que vem ganhando mercado a cada dia. Brevemente, abordaremos outros tópicos mais avançados da programação Delphi/Internet/Intranet.

Andre Campos

Dúvidas e sugestões **sobre o artigo** podem ser enviadas para info@alfamicro.com.br .