Uma produção

Banalsoft

Programação Avançada

Por ***Murphy***

Registro do Windows

Como usar o Registro do Windows no seu programa Delphi.

Os programas possuem cada vez mais opções. E você precisa gravar essas opções em algum lugar. A primeira coisa que lhe vem à cabeça é a criação de um arquivo INI, como no Windows 3.x. O Delphi possui o objeto *TIniFile*, que ajuda a ler e gravar dados nesse arquivo. A não ser que esteja usando o Windows 3.x, essa técnica deve ser repensada. No Windows 95 foi criado o Registro do Windows (*Windows Registry*), cuja finalidade é substituir os vários arquivos INI espalhados pelo Winchester e encapsulá-los em um local centralizado.

Para você ter uma idéia de como ele funciona, dê uma olhada no Editor de Registro, que vem na instalação padrão do Windows (vá em Iniciar|Executar e digite RegEdit). O programa separa as opções de registro em "pastas", conhecidas aqui como "*chaves*". Essas chaves contém dados, chamados de "*valores*". Esses valores possuem um nome, um tipo e um valor específico.

O Delphi encapsula o Registro do Windows através o objeto ***TRegistry***. É através dessa classe que você irá acessar dados do registro.
Para você entender melhor como funciona o objeto, vamos explicar primeiro como funciona o Registro do Windows:

As principais chaves "raízes" são: *HKEY_LOCAL_MACHINE* e *HKEY_CURRENT_USER*. Todas as informações sobre os programas ficam nessas chaves (as outras normalmente possuem informações sobre o Windows).

O InstallShield, por exemplo, instala as informações sobre o nome de usuário e empresa do seu programa em *HKEY_LOCAL_MACHINE\SOFTWARE\Nome da Sua Empresa\Nome Do Programa\Versão*, com os valores *Company* e *Name* (Empresa e Nome do Usuário).

Os programas, por sua vez, usam a chave *HKEY_CURRENT_USER\Software* para armazenar informações de seus programas. E é nessa chave que você deve inserir sua sub-chave e colocar lá as informações necessárias.

Um pequeno exemplo de como utilizar o registro do Windows. Suponhamos que o programa necessite gravar a posição da janela, o seu tamanho e o diretório inicial dos diálogos Abrir e Salvar.

Após a criação do objeto, devemos informar qual chave devemos utilizar, utilizando o método *OpenKey (Chave, PodeCriar);*
onde Chave é o nome da sub-chave e PodeCriar é um valor booleano que permite (ou não) criar a chave caso a mesma não exista.

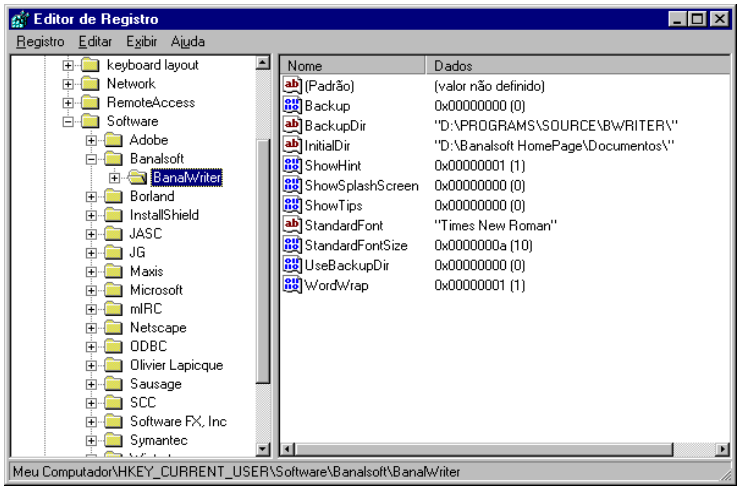

Figura 1: Editor de Registro do Windows.

Na página seguinte há um pedaço de código que faz acesso ao Registro do Windows:

```
{ Este exemplo mostra como funciona como
podemos gravar dados do registro do
Windows utilizando o Delphi 2 ou 3 }

procedure frmMain.GravarRegistro;
const
  Raiz : String = 'Software\Programa';
var
  Registro : TRegistry;
begin
  // Chama o construtor do objeto
  Registro := TRegistry.Create;
{ Abre a chave (se o 2°. Parâmetro
for True, ele cria a chave caso ela
ainda não exista. }
  Registro.OpenKey (Raiz, True);
// Grava as informações do form
  Registro.WriteInteger ('Largura',
Width);
  Registro.WriteInteger ('Altura',
Height);
  Registro.WriteInteger ('Esquerda',
Left);
  Registro.WriteInteger ('Topo', Top);
// Grava as informações das caixas
// Abrir e Salvar.
  Registro.WriteString ('Abrir Inicial',
OpenDialog1.InitialDir);
  Registro.WriteString('Salvar Inicial',
SaveDialog1.InitialDir);
// Fecha a chave e o objeto
  Registro.CloseKey;
  Registro.Free;
end;
```

Após a criação do objeto, deve-se escolher uma chave para armazenas os valores. No caso, "*Software\Programa*", cuja chave raiz é *HKEY_CURRENT_USER*. Note que é para se separar as chaves das sub-chaves utiliza-se o caracter "\", tal como nos diretórios do DOS.

Os métodos *WriteInteger* e *WriteString* são utilizados para gravar valores inteiros e caracteres, respectivamente. A sintaxe básica é:

Registro.*WriteString (*NomeDoValor, Conteúdo);

onde *NomeDoValor* é o nome que você vai dar ao valor dentro da chave, e *Conteúdo* é o conteúdo desse valor.

Para se escrever dados de outros tipos, utilize as funções:

WriteBool (NomeDoValor, Conteúdo); **// Boolean**
WriteBinaryData (NomeDoValor, Conteúdo); **// Valor Binário**
WriteCurrency (NomeDoValor, Conteúdo); **// Currency**
WriteDate (NomeDoValor, Conteúdo); **// TDateTime**
WriteDateTime (NomeDoValor, Conteúdo); **// TDateTime**
WriteFloat (NomeDoValor, Conteúdo); **// Float (Real)**
WriteInteger (NomeDoValor, Conteúdo); **// Integer**
WriteString (NomeDoValor, Conteúdo); **// String**
WriteTime (NomeDoValor, Conteúdo); **// TDateTime**

Sempre use *CloseKey* quando não for precisar do Registro. Isso permite que as opções sejam gravadas permanentemente, evitando que qualquer problema que o computador tenha afete seu programa.

```
{ Este exemplo mostra como funciona como
podemos ler dados do registro do Windows
utilizando o Delphi 2 ou 3 }

procedure frmMain.LerRegistro;
const
  Raiz : String = 'Software\Programa';
var
  Registro : TRegistry;
begin
  // Chama o construtor do objeto
  Registro := TRegistry.Create;
  with Registro do
  begin
// Somente abre se a chave existir
  if OpenKey (Raiz, False) then
// Envia as informações ao form, vendo se os
// valores existem, primeiramente...
  if ValueExists ('Largura') then
    Width := ReadInteger ('Largura');
  if ValueExists ('Altura') then
    Height := ReadInteger ('Altura');
  if ValueExists ('Esquerda') then
    Left := ReadInteger ('Esquerda');
  if ValueExists ('Topo') then
    Top := ReadInteger ('Topo');
// Envia as informações para as caixas
// Abrir e Salvar.
 OpenDialog1.InitialDir := ReadString ('Abrir
Inicial');
 SaveDialog1.InitialDir := ReadString
('Salvar Inicial');
// Fecha a chave e o objeto
  Registro.CloseKey;
  Registro.Free;
end;
```

Os métodos Read*Integer* e Read*String* funcionam praticamente da mesma maneira que seus correspondentes de escrita. A diferença é que ao invés de passar o valor *Conteúdo*, eles retornam o valor armazenado. Os correspondentes dos outros tipos são:

ReadBool (NomeDoValor) : Boolean;
*ReadBinaryData (NomeDoValor ; **var** Buffer ; TamBuffer : Integer): Integer;*
ReadCurrency (NomeDoValor) : Currency;
ReadDate (NomeDoValor) : TDateTime;
ReadDateTime (NomeDoValor) : TDateTime;
ReadFloat (NomeDoValor) : Double;
ReadInteger (NomeDoValor) : Integer;

ReadString (NomeDoValor) : String;
ReadTime (NomeDoValor) : TDateTime;