Serialize application settings

Store application settings in a Delphi class and serialize it to a binary file:

```
unit SettingsU;

interface

uses Classes;

{$M+}

type
  TCustomSettings = class
  public
    procedure LoadFromStream(const Stream: TStream);
    procedure LoadFromFile(const FileName: string);
    procedure SaveToStream(const Stream: TStream);
    procedure SaveToFile(const FileName: string);
  end;

  TSettings = class(TCustomSettings)
  private
    FPropertyString: string;
    FPropertyDate: TDateTime;
    FPropertyInt: Integer;
  published
    property PropertyInt: Integer
      read FPropertyInt write FPropertyInt;
    property PropertyString: string
      read FPropertyString write FPropertyString;
    property PropertyDate: TDateTime
      read FPropertyDate write FPropertyDate;
  end;

var
  Settings: TSettings;

implementation

uses TypInfo, Sysutils;

{ TSettings }

procedure TCustomSettings.LoadFromFile(const FileName: string);
var
  Stream: TStream;
begin
  Stream:= TFileStream.Create(FileName, fmOpenRead or fmShareDenyWrite);
  try
    LoadFromStream(Stream);
  finally
    Stream.Free;
  end;
end;

procedure TCustomSettings.LoadFromStream(const Stream: TStream);
var
  Reader: TReader;
  PropName, PropValue: string;
begin
  Reader:= TReader.Create(Stream, $FFF);
  Stream.Position:= 0;
  Reader.ReadListBegin;

  while not Reader.EndOfList do
  begin
    PropName:= Reader.ReadString;
    PropValue:= Reader.ReadString;
    SetPropValue(Self, PropName, PropValue);
  end;
```

```
  FreeAndNil(Reader);
end;

procedure TCustomSettings.SaveToFile(const FileName: string);
var
  Stream: TStream;
begin
  Stream:= TFileStream.Create(FileName, fmCreate);
  try
    SaveToStream(Stream);
  finally
    Stream.Free;
  end;
end;

procedure TCustomSettings.SaveToStream(const Stream: TStream);
var
  PropName, PropValue: string;
  cnt: Integer;
  lPropInfo: PPropInfo;
  lPropCount: Integer;
  lPropList: PPropList;
  lPropType: PPTypeInfo;
  Writer: TWriter;
begin
  lPropCount:= GetPropList(PTypeInfo(ClassInfo), lPropList);
  Writer:= TWriter.Create(Stream, $FFF);
  Stream.Size:= 0;
  Writer.WriteListBegin;
  for cnt:= 0 to lPropCount - 1 do
  begin
    lPropInfo:= lPropList^[cnt];
    lPropType:= lPropInfo^.PropType;
    if lPropType^.Kind = tkMethod then Continue;

    PropName:= lPropInfo.Name;
    PropValue:= GetPropValue(Self, lPropInfo);
    Writer.WriteString(PropName);
    Writer.WriteString(PropValue);
  end;

  Writer.WriteListEnd;
  FreeAndNil(Writer);
end;

initialization

Settings:= TSettings.Create;

finalization

FreeAndNil(Settings);

end.
```

Author:	Shlomo Abuisak
Contributor:	Shlomo Abuisak
Added:	2009-11-05
Last updated:	2009-11-05