```pascal
unit AppSettings;

{$mode objfpc}{$H+}

//========================================================================================
//
// Unit : AppSettings.pas
//
// Description :
//
// Called By :  AppInit : Initialize
//              Main  : TfrmMain.mnuSettingsDIrectoriesClick
//                      TfrmMain.mnuSettingsDatabasesClick
//              SuppliersTable : frmSuppliersTable.CreateSuppliersTable
//
// Calls :  HUConstants
//
// Ver. : 1.0.0
//
// Date : 22 Jan 2020
//
//========================================================================================

interface

uses
  Buttons, Classes, ComCtrls, Controls, Dialogs, FileUtil, Forms, Graphics,
  INIFiles, sqlite3conn, sqldb, db, StdCtrls, DBGrids, DBCtrls, Grids, SysUtils,
  Types, windirs, STRUtils,
  //App Units
  // HULibrary units
  HUConstants, HUMessageBoxes, HURegistration;

type

  { TfrmSettings }

  TfrmSettings = class(TForm)
    bbtOkClose: TBitBtn;
    DBTableDataSource: TDataSource;
    DBTableQuery: TSQLQuery;
    dsRegistrationSettingsTable: TDataSource;
    dsApplicationSettingsTable: TDataSource;
    dbeFirstName: TDBEdit;
    edtBackupsDirectory: TEdit;
    edtAppDataDirectory: TEdit;
    edtLogbooksDirectory: TEdit;
    edtSettingsDirectory: TEdit;
    edtApplicationDirectory: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    pcSettings: TPageControl;
    pgDirectories: TTabSheet;
    pgSettingsDB: TTabSheet;
    DBConnection: TSQLite3Connection;
    sqlqApplicationSettingsTable: TSQLQuery;
    sqlqRegistrationSettingsTable: TSQLQuery;
    DBTransaction: TSQLTransaction;
    pgApplicationSettings: TTabSheet;
    pgRegistrationSettings: TTabSheet;
    StatusBar1: TStatusBar;
    strgrdApplicationSettings: TStringGrid;
    strgrdRegistrationSettings: TStringGrid;
    procedure bbtCancelCloseClick(Sender: TObject);
    procedure bbtOkCloseClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);
    procedure FormCreate(Sender: TObject);
    procedure FormShow(Sender: TObject);
    function CreateApplicationDataBase: boolean;
    function LoadApplicationDatabase: boolean;
    function SaveApplicationDataBase: Boolean;

  private
    //====================================================================================
    // The following Properties are in fact constants that once initialized will not be
    // changed during program execution.
    //====================================================================================
      // Application Elements
    fAppVersion : string;
    fAppFullFilePathName : string;
    fAppFilePath : string;
    fAppFullFileName : string;
    fAppFileName : string;
    fAppFileExt : string;
    fAppUserDirectory : string;
    fAppDataDirectory : string;
    fAppSettingsDirectory : string;
    fAppLogbooksDirectory : string;
    fAppBackupsDirectory : string;
    //====================================================================================
    // The following Properties are in fact variables that once initialized may be
    // changed during program execution and are saved in the AppSettings database.
    //====================================================================================
    fAppSettingsInitialPageName : string;
    fAppSettingsInitialPageNum : string;
    fAppDatabaseName : string;

    //====================================================================================
    // The following Properties are in fact constants that once initialized will not be
    // changed during program execution.
    //====================================================================================
    function GetAppVersion : string;
    procedure SetAppVersion(Version : string);
    function GetAppFullFilePathName : string;
    procedure SetAppFullFilePathName(PathName : string);
    function GetAppFilePath : string;
    procedure SetAppFilePath(Path : string);
    function GetAppFullFileName : string;
    procedure SetAppFullFileName(FName : string);
    function GetAppFileName : string;
    procedure SetAppFileName(FName : string);
    function GetAppFileExt : string;
    function GetAppUserDirectory : string;
    procedure SetAppUserDirectory(Dir : string);
    function GetAppDataDirectory : string;
    procedure SetAppDataDirectory(Dir : string);
    function GetAppSettingsDirectory : string;
    procedure SetAppSettingsDirectory(Dir : string);
    function GetAppLogbooksDirectory : string;
    procedure SetAppLogbooksDirectory(Dir : string);
    function GetAppBackupsDirectory : string;
    procedure SetAppBackupsDirectory(Dir : string);
    //====================================================================================
    // The following Properties are in fact variables that once initialized may be
    // changed during program execution and are saved in the AppSettings database.
    //====================================================================================
    function GetAppSettingsInitialPageName : string;
    procedure SetAppSettingsInitialPageName(PageName : string);
    function GetAppSettingsInitialPageNum : string;
    procedure SetAppSettingsInitialPageNum(PageNum : string);
    function GetAppDatabaseName : string;
    procedure SetAppDatabaseName(DBName : string);

  public
    //====================================================================================
    // The following Properties are in fact constants that once initialized will not be
    // changed during program execution.
    //====================================================================================
    property pAppVersion : string
                           read GetAppVersion
                           write SetAppVersion;
    property pAppFullFilePathName : string
                                    read GetAppFullFilePathName
                                    write SetAppFullFilePathName;
    property pAppFilePath : string read GetAppFilePath
                                   write SetAppFilePath;
    property pAppFullFileName : string
                                read GetAppFullFileName
                                write SetAppFullFileName;
    property pAppFileName : string
                            read GetAppFileName
                            write SetAppFileName;
    property pAppFileExt : string
                           read GetAppFileExt;
    property pAppUserDirectory : string
                                 read GetAppUserDirectory
                                 write SetAppUserDirectory;
    property pAppDataDirectory : string
                                 read GetAppDataDirectory
                                 write SetAppDataDirectory;
    property pAppSettingsDirectory : string
                                     read GetAppSettingsDirectory
                                     write SetAppSettingsDirectory;
    property pAppLogbooksDirectory : string
                                     read GetAppLogbooksDirectory
                                     write SetAppLogbooksDirectory;
    property pAppBackupsDirectory : string
                                    read GetAppBackupsDirectory
                                    write SetAppBackupsDirectory;
    property pAppDatabaseName : string
                                read GetAppDatabaseName
                                write SetAppDatabaseName;

    //====================================================================================
    // The following Properties are in fact variables that once initialized may be
    // changed during program execution and are saved in the AppSettings database.
    //=========================================================================
    property pAppSettingsInitialPageName : string
                                  read GetAppSettingsInitialPageName
                                 write SetAppSettingsInitialPageName;
    property pAppSettingsInitialPageNum : string
                                  read GetAppSettingsInitialPageNum
                                 write SetAppSettingsInitialPageNum;

  end;// TfrmSettings

var
  frmSettings: TfrmSettings;

implementation

{$R *.lfm}

//============================================================================
//          PRIVATE CONSTANTS
//============================================================================
const

  // Directories
  cstrAppFullFileName = 'RVMasterLog.exe';
  cstrSettingsDirectoryName = 'Settings';
  cstrLogbooksDirectoryName = 'Logbooks';
  cstrBackupsDirectoryName = 'Backups';
  cstrAppDataDirectoryName = 'AppData';

  // Databases
  cstrApplicationDBName = 'RVMApplicationDB.sl3';
  cstrApplicationSettingsTableName = 'ApplicationSettingsTable';

  // Properties
  cstrpAppSettingsInitialPageName = 'pAppSettingsInitialPageName';
  cstrpAppSettingsInitialPageNum = 'pAppSettingsInitialPageNum';

  // frmSettings
  straryPageNames : array[0..3] of string = ('Directories', 'Application Settings',
                                            'Registration Settings', 'SettingsDB');

//============================================================================
//          PRIVATE VARIABLES
//============================================================================
var
  vintAppSettingsInitialDirectory : integer;

//============================================================================
//          PRIVATE ROUTINES
//============================================================================

//============================================================================
//          PUBLIC ROUTINES
//============================================================================
function TfrmSettings.CreateApplicationDataBase: Boolean;
var
  vstrTstr : String;
begin

```

```
showmessage('Creating ApplicationDataBase');

  DBConnection.Close; // Ensure any connection is closed when we start

  Result := True;

    //==========================================
    // Create the Default database and tables
    //==========================================
  try

    //=========================
    //  Create the Database
    //=========================
    DBConnection.Open;
    DBTransaction.Active := true;

    //==========================================
    // Create the "ApplicationSettingsTable"
    //==========================================

// showmessage('Create ApplicationSettingsTable');

    DBConnection.ExecuteDirect('CREATE TABLE "ApplicationSettingsTable"('+
                                    ' "Property" String PRIMARY KEY,'+
                                    ' "Value" String );');

// showmessage('Create ApplicationSettingsIndex');

    // Creating an index based upon Property in the ApplicationSettingsTable
    DBConnection.ExecuteDirect('CREATE UNIQUE INDEX ' +
                                    ' "ApplicationSettingsTable_Property_idx"' +
                                    ' ON "ApplicationSettingsTable"( "Property" );');

    //==========================================
    // Create the "RegistrationSettingsTable"
    //==========================================

// showmessage('Create RegistrationSettingsTable');

    DBConnection.ExecuteDirect('CREATE TABLE "RegistrationSettingsTable"('+
                                    ' "Property" String PRIMARY KEY,'+
                                    ' "Value" String );');

// showmessage('Create RegistrationSettings Index');


    DBConnection.ExecuteDirect('CREATE UNIQUE INDEX ' +
                                    ' "RegistrationSettingsTable_Property_idx"' +
                                    ' ON "RegistrationSettingsTable"( "Property" );');
    DBTransaction.Commit;

    //=========================
    // Add the User Adaptble Property Records with Initial Default values.
    //=============================

    //================================
    // Application Settings properties
    //================================
    DBConnection.ExecuteDirect('INSERT INTO ApplicationSettingsTable VALUES' +
                                     ' ("pAppSettingsInitialPageName", "Application Settings")');
    DBConnection.ExecuteDirect('INSERT INTO ApplicationSettingsTable VALUES' +
                                    ' ("pAppSettingsInitialPageNum", "1")');
    //==========================
    // HURegistration properties
    //==========================
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegFirstName", " ")');
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegLastName", " ")');
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegEMailaddress", " ")');
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegCallsign", " ")');
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegKey", " ")');
    DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
                                     ' ("pRegUserID", " ")');

    //  Additional records go here

    //=========================
    // Commit the additions
    //=========================

    DBTransaction.Commit;

    except
      ShowMessage('Unable to Create new Database');
      Result := False;
  end;// Try to Create the Default database and tables

  //=======================
  //  Database Created
  //=======================

  DBTransaction.Active := False;
  DBConnection.Close;

  showmessage('Settings DataBase Created');

end;// function TfrmSettings.CreateSettingsDataBase

//============================================================================
function TfrmSettings.LoadApplicationDatabase : boolean;
var
  vstrTStr : string;
  vintRecNr : integer;

begin

  showmessage('LoadApplicationDatabase');

  Result := True;

  try {LoadApplicationDatabase}

//*****    showmessage('Opening DBConnection');

```

```
    if not DBConnection.Connected then
      DBConnection.Open;

//    showmessage('DBConnection Open');

    if not DBConnection.Connected then
    begin
      showmessage('Error connecting to the Database. Aborting data loading');
      Result := False;
      Exit;
    end;// if not DBConnection.Connected

    //==================================================
    // Load the Application Settings Table properties
    //==================================================
    sqlqApplicationSettingsTable.SQL.Text :=
      'select ' +
      '  e.Property, ' +
      '  e.Value ' +
      'from ApplicationSettingsTable e';

    DBTransaction.StartTransaction;

//    showmessage('DBTransaction.StartTransaction');

    sqlqApplicationSettingsTable.Open;

//    showmessage('sqlqApplicationSettingsTable.Open');

    // Get pAppSettingsInitialPageName
    //    showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo   )));
    pAppSettingsInitialPageName := sqlqApplicationSettingsTable.Fields[1].AsString;

    // Get pAppSettingsInitialPageNum
    sqlqApplicationSettingsTable.NEXT;
//    showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo   )));
    pAppSettingsInitialPageNum := sqlqApplicationSettingsTable.Fields[1].AsString;
//    showmessage('pAppSettingsInitialPageNum = ' + pAppSettingsInitialPageNum);

    //==================================================
    // Application Settings Table properties  Loaded
    //==================================================
//    showmessage('ApplicationSettingsTable.EOF');
    sqlqApplicationSettingsTable.Close;


    //==================================================
    // Load the Registration Table properties
    //==================================================
    sqlqRegistrationSettingsTable.SQL.Text :=
      'select ' +
      '  e.Property, ' +
      '  e.Value ' +
      'from RegistrationSettingsTable e';

    sqlqRegistrationSettingsTable.Open;
//    showmessage('sqlqRegistrationSettingsTable Open');

    // Get pRegFirstName
//    showmessage('Record = ' + (IntToStr(sqlqRegistrationSettingsTable.RecNo   )));
```

```
    dlgHURegistration.pRegFirstName := sqlqRegistrationSettingsTable.Fields[1].AsString;

    // Get pRegLastName
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegLastName := sqlqRegistrationSettingsTable.Fields[1].AsString;

    // Get pRegEmailAddress
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegEmailAddress := sqlqRegistrationSettingsTable.Fields[1].AsString;

     // Get pRegCallsign
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegCallsign := sqlqRegistrationSettingsTable.Fields[1].AsString;

         // Get pRegKey
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegKey := sqlqRegistrationSettingsTable.Fields[1].AsString;

        // Get pRegUserID
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegUserID := sqlqRegistrationSettingsTable.Fields[1].AsString;

    //==================================================
    // Registration Table properties  Loaded
    //==================================================
    showmessage('RegistrationTable.EOF');
    sqlqRegistrationSettingsTable.Close;

  except

    on D: EDatabaseError do
    begin
      MessageDlg('Error', 'A Database error has occured. Technical error message: ' +
                          D.Message, mtError, [mbOK], 0);
      Result := False;
    end;// on D: EDatabaseEorror

  end;// Try {LoadApplicationDatabase}

  DBTransaction.Active := False;
  DBConnection.Close;

end;// function TfrmSettings.LoadApplicationDatabase

//============================================================================
function TfrmSettings.SaveApplicationDatabase : Boolean;
{
var
  vstrTStr : string;
  vintRecNr : integer;
}
begin
{
  showmessage('SaveApplicationDatabase');

  Result := True;

  try {LoadApplicationDatabase}

//*****    showmessage('Opening DBConnection');
```

```

    if not DBConnection.Connected then
      DBConnection.Open;

//    showmessage('DBConnection Open');

    if not DBConnection.Connected then
    begin
      showmessage('Error connecting to the Database. Aborting data loading');
      Result := False;
      Exit;
    end;// if not DBConnection.Connected

    //================================================
    // Load the Application Settings Table properties
    //================================================
    sqlqApplicationSettingsTable.SQL.Text :=
      'select ' +
      '  e.Property, ' +
      '  e.Value ' +
      'from ApplicationSettingsTable e';

    DBTransaction.StartTransaction;

//    showmessage('DBTransaction.StartTransaction');

    sqlqApplicationSettingsTable.Open;

//    showmessage('sqlqApplicationSettingsTable.Open');

    // Get pAppSettingsInitialPageName
    //    showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo   )));
    pAppSettingsInitialPageName := sqlqApplicationSettingsTable.Fields[1].AsString;

    // Get pAppSettingsInitialPageNum
    sqlqApplicationSettingsTable.NEXT;
//    showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo   )));
    pAppSettingsInitialPageNum := sqlqApplicationSettingsTable.Fields[1].AsString;
//    showmessage('pAppSettingsInitialPageNum = ' + pAppSettingsInitialPageNum);

    //================================================
    // Application Settings Table properties  Loaded
    //================================================
//    showmessage('ApplicationSettingsTable.EOF');
    sqlqApplicationSettingsTable.Close;


    //================================================
    // Load the Registration Table properties
    //================================================
    sqlqRegistrationSettingsTable.SQL.Text :=
      'select ' +
      '  e.Property, ' +
      '  e.Value ' +
      'from RegistrationSettingsTable e';

    sqlqRegistrationSettingsTable.Open;
//    showmessage('sqlqRegistrationSettingsTable Open');

    // Get pRegFirstName
//    showmessage('Record = ' + (IntToStr(sqlqRegistrationSettingsTable.RecNo   )));
    dlgHURegistration.pRegFirstName := sqlqRegistrationSettingsTable.Fields[1].AsString;

    // Get pRegLastName
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegLastName := sqlqRegistrationSettingsTable.Fields[1].AsString;

    // Get pRegEmailAddress
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegEmailAddress := sqlqRegistrationSettingsTable.Fields[1].AsString;

     // Get pRegCallsign
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegCallsign := sqlqRegistrationSettingsTable.Fields[1].AsString;

         // Get pRegKey
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegKey := sqlqRegistrationSettingsTable.Fields[1].AsString;

        // Get pRegUserID
    sqlqRegistrationSettingsTable.NEXT;
    dlgHURegistration.pRegUserID := sqlqRegistrationSettingsTable.Fields[1].AsString;

    //==============================================
    // Registration Table properties  Loaded
    //==============================================
    showmessage('RegistrationTable.EOF');
    sqlqRegistrationSettingsTable.Close;

  except

    on D: EDatabaseError do
    begin
      MessageDlg('Error', 'A Database error has occured. Technical error message: ' +
                          D.Message, mtError, [mbOK], 0);
      Result := False;
    end;// on D: EDatabaseEorror

  end;// Try {LoadApplicationDatabase}

  DBTransaction.Active := False;
  DBConnection.Close;
}
end;// function TfrmSettings.SaveSettingsDataBase

//==========================================================================================
//          PROPERTY ROUTINES
//==========================================================================================
function TfrmSettings.GetAppVersion: string;
begin
   Result := fAppVersion;
end;// function TfrmSettings.GetVersion

//------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppVersion(Version: string);
begin
    fAppVersion := Version;
end;// procedure TfrmSettings.SetAppVersion

//==========================================================================================
function TfrmSettings.GetAppFullFilePathName: string;
begin
   Result := fAppFullFilePathName;
end;// function TfrmSettings.GetAppFullFilePathName

//----------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppFullFilePathName(PathName: string);
begin
    fAppFullFilePathName := PathName;
end;// procedure TfrmSettings.SetAppFullFilePathName

//========================================================================================
function TfrmSettings.GetAppFilePath: string;
begin
   Result := fAppFilePath;
end;// function TfrmSettings.GetAppFullFilePath

//----------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppFilePath(Path: string);
begin
    fAppFilePath := Path;
end;// procedure TfrmSettings.SetAppFullFilePath

//========================================================================================
function TfrmSettings.GetAppFullFileName: string;
begin
   Result := fAppFullFileName;
end;// function TfrmSettings.GetAppFullFileNamee

//----------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppFullFileName(FName: string);
begin
    fAppFullFileName := cstrAppFullFileName;
end;// procedure TfrmSettings.SetAppFullFileName

//========================================================================================
function TfrmSettings.GetAppFileName: string;
begin
   Result := fAppFileName;
end;// function TfrmSettings.GetAppFileNamee

//----------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppFileName(FName: string);
begin
    fAppFileName := FName;
end;// procedure TfrmSettings.SetAppFileName

//========================================================================================
function TfrmSettings.GetAppFileExt: string;
begin
   Result := fAppFileExt;
end;// function TfrmSettings.GetAppFileExt

//========================================================================================
function TfrmSettings.GetAppUserDirectory: string;
begin
   Result := fAppUserDirectory;
end;// procedure TfrmSettings.GetAppUserDirectory

//----------------------------------------------------------------------------------------
```

```
procedure TfrmSettings.SetAppUserDirectory(Dir: string);
begin
    fAppUserDirectory := Dir;
end;// procedure TfrmSettings.SetAppUserDataDirectory

//===========================================================================================
function TfrmSettings.GetAppDataDirectory: string;
begin
   Result := fAppDataDirectory;
end;// procedure TfrmSettings.GetAppDirectory

//-------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppDataDirectory(Dir: string);
begin
    fAppDataDirectory := Dir;
end;// procedure TfrmSettings.SetAppDataDirectory

//===========================================================================================
function TfrmSettings.GetAppSettingsDirectory: string;
begin
   Result := fAppSettingsDirectory;
end;// procedure TfrmSettings.GetAppSettingsDirectory

//-------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppSettingsDirectory(Dir: string);
begin
    fAppSettingsDirectory := Dir;
end;// procedure TfrmSettings.SetAppSettingsDirectory

//===========================================================================================
function TfrmSettings.GetAppLogbooksDirectory: string;
begin
   Result := fAppLogbooksDirectory;
end;// procedure TfrmSettings.GetAppLogbooksDirectory

//-------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppLogbooksDirectory(Dir: string);
begin
    fAppLogbooksDirectory := Dir;
end;// procedure TfrmSettings.SeApptLogbooksDirectory

//===========================================================================================
function TfrmSettings.GetAppBackupsDirectory: string;
begin
   Result := fAppBackupsDirectory;
end;// procedure TfrmSettings.GetAppBackupsDirectory

//-------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppBackupsDirectory(Dir: string);
begin
    fAppBackupsDirectory := Dir;
end;// procedure TfrmSettings.SetAppBackupsDirectory

//===========================================================================================
function TfrmSettings.GetAppSettingsInitialPageName: string;
begin
   Result := fAppSettingsInitialPageName;
end;// procedure TfrmSettings.GetAppSettingsInitialPageName

//-------------------------------------------------------------------------------------------
```

```
procedure TfrmSettings.SetAppSettingsInitialPageName(PageName: string);
begin
    fAppSettingsInitialPageName := PageName;
end;// procedure TfrmSettings.SetAppSettingsInitialPageName

//============================================================================================
function TfrmSettings.GetAppSettingsInitialPageNum: string;
begin
   Result := fAppSettingsInitialPageNum ;
end;// procedure TfrmSettings.GetAppSettingsInitialPageNum

//--------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppSettingsInitialPageNum(PageNum: string);
begin
    fAppSettingsInitialPageNum := PageNum;
end;// procedure TfrmSettings.SetAppSettingsInitialPageNum

//============================================================================================
function TfrmSettings.GetAppDatabaseName: string;
begin
   Result := fAppDatabaseName;
end;// procedure TfrmSettings.GetAppDatabaseName

//--------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppDatabaseName(DBName: string);
begin
    fAppDatabaseName := DBName;
end;// procedure TfrmSettings.SetAppDatabaseName

{//============================================================================================
function TfrmSettings.GetAppOwnerFirstName: string;
begin
   Result := fAppOwnerFirstName;
end;// procedure TfrmSettings.GetAppOwnerFirstName

//--------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppOwnerFirstName(FirstName: string);
begin
    fAppOwnerFirstName := FirstName;
end;// procedure TfrmSettings.SetAppOwnerFirstName

//============================================================================================
function TfrmSettings.GetAppOwnerLastName: string;
begin
   Result := fAppOwnerLastName;
end;// procedure TfrmSettings.GetAppOwnerLastName

//--------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppOwnerLastName(LastName: string);
begin
    fAppOwnerLastName := LastName;
end;// procedure TfrmSettings.SetAppOwnerLastName

//============================================================================================
function TfrmSettings.GetAppOwnerCallsign: string;
begin
   Result := fAppOwnerCallsign;
end;// procedure TfrmSettings.GetAppOwnerCallsign

//--------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppOwnerCallsign(Callsign: string);
begin
    fAppOwnerCallsign := Callsign;
end;// procedure TfrmSettings.SetAppOwnerCallsign

//==========================================================================================
function TfrmSettings.GetAppOwnerEmailAddress: string;
begin
   Result := fAppOwnerEmailAddress;
end;// procedure TfrmSettings.GetAppOwnerEmailAddress

//------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppOwnerEmailAddress(EmailAddress: string);
begin
    fAppOwnerEmailAddress := EMailAddress;
end;// procedure TfrmSettings.SetAppOwnerEmailAddress

//==========================================================================================
function TfrmSettings.GetAppOwnerID: string;
begin
   Result := fAppOwnerID;
end;// procedure TfrmSettings.GetAppUserID

//------------------------------------------------------------------------------------------
procedure TfrmSettings.SetAppOwnerID(OwnerID: string);
begin
    fAppOwnerID := OwnerID;
end;// procedure TfrmSettings.SetAppOwnerID }

//==========================================================================================
//          MENU ROUTINES
//==========================================================================================

//==========================================================================================
//          COMMAND BUTTON ROUTINES
//==========================================================================================
procedure TfrmSettings.bbtCancelCloseClick(Sender: TObject);
begin

end;//  procedure TfrmSettings.bbtCancelClick

//------------------------------------------------------------------------------------------
procedure TfrmSettings.bbtOkCloseClick(Sender: TObject);
begin

end;//  procedure TfrmSettings.bbtOkClick

//==========================================================================================
//          CONTROL ROUTINES
//==========================================================================================

//==========================================================================================
//          FILE ROUTINES
//==========================================================================================

//==========================================================================================
//          FORM ROUTINES
//==========================================================================================
procedure TfrmSettings.FormCreate(Sender: TObject);
begin
```

```

  //=====================================================
  // Initialize the Application Directories
  //=====================================================

  pAppFileName := Copy (cstrAppFullFileName, 1, 12);
  pAppFilePath := GetCurrentDir;
  pAppUserDirectory := (GetWindowsSpecialDir(CSIDL_PERSONAL)) + pAppFileName;

    // The following directories are created by the INNO Setup Script
    pAppSettingsDirectory := pAppUserDirectory + '\' + cstrSettingsDirectoryName;
    pAppLogbooksDirectory := pAppUserDirectory + '\' + cstrLogbooksDirectoryName;
    pAppBackupsDirectory := pAppUserDirectory + '\' + cstrBackupsDirectoryName;
    pAppDataDirectory := pAppUserDirectory + '\' + cstrApPDataDirectoryName;
    pAppDatabaseName := pAppDataDirectory + '\' + cstrApplicationDBName;

    //=====================================================
    // Initialize the Data Components
    //=====================================================
    DBConnection.DatabaseName := pAppDatabaseName;
    DBConnection.Transaction := DBTransaction;
    sqlqApplicationSettingsTable.DataBase := DBConnection;
    sqlqApplicationSettingsTable.DataSource := dsApplicationSettingsTable;
    sqlqRegistrationSettingsTable.DataBase := DBConnection;
    sqlqRegistrationSettingsTable.DataSource := dsRegistrationSettingsTable;

    //=====================================================
    // Initialize the Controls
    //=====================================================

    // PageControl properties
    pgDirectories.Caption:=straryPageNames[0];
    pgApplicationSettings.Caption:=straryPageNames[1];
    pgRegistrationSettings.Caption:=straryPageNames[2];
    pgSettingsDB.Caption:=straryPageNames[3];

    // strgApplicationSettings properties
    strgrdApplicationSettings.AutoFillColumns := True;
    strgrdApplicationSettings.AutoSizeColumns;

    // strgRegistrationSettings properties
    strgrdRegistrationSettings.AutoFillColumns := True;
    strgrdRegistrationSettings.AutoSizeColumns;

end;// procedure TfrmSettings.FormCreate

//--------------------------------------------------------------------------------
procedure TfrmSettings.FormShow(Sender: TObject);
var
  vstrTstr : string;
  vintTInt : Integer;
begin

  //======================
  // initialize form controls
  //======================

  // Command Buttons

  //  Directories Page
  edtApplicationDirectory.Text:= pAppFilePath;
  edtSettingsDirectory.Text:=pAppSettingsDirectory;
  edtLogbooksDirectory.Text:=pAppLogbooksDirectory;
  edtBackupsDirectory.Text := pAppBackupsDirectory;
  edtAppDataDirectory.Text := pAppDataDirectory;

  //  Application Settings Page

  // strgApplicationSettings
  strgrdApplicationSettings.Cells[0,1] := 'pAppSettingsInitialPageNum';
  strgrdApplicationSettings.Cells[1,1] := pAppSettingsInitialPageNum;
  strgrdApplicationSettings.Cells[0,2] := 'pAppSettingsInitialPageName';
  strgrdApplicationSettings.Cells[1,2] := pAppSettingsInitialPageName;

  // strgRegistrationSettings
  strgrdRegistrationSettings.Cells[0,1] := 'dlgHURegistration.pRegFirstName';
  strgrdRegistrationSettings.Cells[1,1] := dlgHURegistration.pRegFirstName;
  strgrdRegistrationSettings.Cells[0,2] := 'dlgHURegistration.pRegLastName';
  strgrdRegistrationSettings.Cells[1,2] := dlgHURegistration.pRegLastName;
  strgrdRegistrationSettings.Cells[0,3] := 'dlgHURegistration.pRegEmailAddress';
  strgrdRegistrationSettings.Cells[1,3] := dlgHURegistration.pRegEmailAddress;
  strgrdRegistrationSettings.Cells[0,4] := 'dlgHURegistration.pRegCallsign';
  strgrdRegistrationSettings.Cells[1,4] := dlgHURegistration.pRegCallsign;
  strgrdRegistrationSettings.Cells[0,5] := 'dlgHURegistration.pRegKey';
  strgrdRegistrationSettings.Cells[1,5] := dlgHURegistration.pRegKey;
  strgrdRegistrationSettings.Cells[0,6] := 'dlgHURegistration.pRegUserID';
  strgrdRegistrationSettings.Cells[1,6] := dlgHURegistration.pRegUserID;

end;// procedure TfrmAppSetup.FormShow

//------------------------------------------------------------------------------------------
procedure TfrmSettings.FormClose(Sender: TObject; var CloseAction: TCloseAction);
begin






  DBConnection.Connected := False;
end;// procedure TfrmAppSetup.FormClose

//==========================================================================================
end.// unit AppSettings

```