

```

1: unit AppSettings;
2:
3: {$mode objfpc}{$H+}
4:
5: //=====
6: //
7: // Unit : AppSettings.pas
8: //
9: // Description :
10: //
11: // Called By :   AppInit : Initialize
12: //              Main  : TfrmMain.mnuSettingsDirectoriesClick
13: //              TfrmMain.mnuSettingsDatabasesClick
14: //              SuppliersTable : frmSuppliersTable.CreateSuppliersTable
15: //
16: // Calls :   HUConstants
17: //
18: // Ver. : 1.0.0
19: //
20: // Date : 22 Jan 2020
21: //
22: //=====
23:
24: interface
25:
26: uses
27:   Buttons, Classes, ComCtrls, Controls, Dialogs, FileUtil, Forms, Graphics,
28:   INIFiles, sqlite3conn, sqldb, db, StdCtrls, DBGrids, DBCtrls, Grids, SysUtils,
29:   Types, windirs, STRUtils,
30:   //App Units
31:   // HULibrary units
32:   HUConstants, HUMessageBoxes, HURegistration;
33:
34: type
35:
36:   { TfrmSettings }
37:
38:   TfrmSettings = class(TForm)
39:     bbtOkClose: TBitBtn;
40:     DBTableDataSource: TDataSource;
41:     DBTableQuery: TSQLQuery;
42:     dsRegistrationSettingsTable: TDataSource;
43:     dsApplicationSettingsTable: TDataSource;
44:     dbeFirstName: TDBEdit;
45:     edtBackupsDirectory: TEdit;
46:     edtAppDataDirectory: TEdit;
47:     edtLogbooksDirectory: TEdit;
48:     edtSettingsDirectory: TEdit;
49:     edtApplicationDirectory: TEdit;
50:     Label1: TLabel;
51:     Label2: TLabel;
52:     Label3: TLabel;
53:     Label4: TLabel;
54:     Label5: TLabel;
55:     Label6: TLabel;
56:     Label7: TLabel;
57:     pcSettings: TPageControl;
58:     pgDirectories: TTabSheet;
59:     pgSettingsDB: TTabSheet;
60:     DBConnection: TSQlite3Connection;

```

```

61:   sqlqApplicationSettingsTable: TSQLQuery;
62:   sqlqRegistrationSettingsTable: TSQLQuery;
63:   DBTransaction: TSQLTransaction;
64:   pgApplicationSettings: TTabSheet;
65:   pgRegistrationSettings: TTabSheet;
66:   StatusBar1: TStatusBar;
67:   strgrdApplicationSettings: TStringGrid;
68:   strgrdRegistrationSettings: TStringGrid;
69:   procedure bbtCancelCloseClick(Sender: TObject);
70:   procedure bbtOkCloseClick(Sender: TObject);
71:   procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);
72:   procedure FormCreate(Sender: TObject);
73:   procedure FormShow(Sender: TObject);
74:   function CreateApplicationDataBase: boolean;
75:   function LoadApplicationDatabase: boolean;
76:   function SaveApplicationDataBase: Boolean;
77:
78: private
79:   //=====
80:   // The following Properties are in fact constants that once initialized will not be
81:   // changed during program execution.
82:   //=====
83:   // Application Elements
84:   fAppVersion : string;
85:   fAppFullFilePathName : string;
86:   fAppFilePath : string;
87:   fAppFullFileName : string;
88:   fAppFileName : string;
89:   fAppFileExt : string;
90:   fAppUserDirectory : string;
91:   fAppDataDirectory : string;
92:   fAppSettingsDirectory : string;
93:   fAppLogbooksDirectory : string;
94:   fAppBackupsDirectory : string;
95:   //=====
96:   // The following Properties are in fact variables that once initialized may be
97:   // changed during program execution and are saved in the AppSettings database.
98:   //=====
99:   fAppSettingsInitialPageName : string;
100:  fAppSettingsInitialPageNum : string;
101:  fAppDatabaseName : string;
102:
103:  //=====
104:  // The following Properties are in fact constants that once initialized will not be
105:  // changed during program execution.
106:  //=====
107:  function GetAppVersion : string;
108:  procedure SetAppVersion(Version : string);
109:  function GetAppFullFilePathName : string;
110:  procedure SetAppFullFilePathName(PathName : string);
111:  function GetAppFilePath : string;
112:  procedure SetAppFilePath(Path : string);
113:  function GetAppFullFileName : string;
114:  procedure SetAppFullFileName(FName : string);
115:  function GetAppFileName : string;
116:  procedure SetAppFileName(FName : string);
117:  function GetAppFileExt : string;
118:  function GetAppUserDirectory : string;
119:  procedure SetAppUserDirectory(Dir : string);
120:  function GetAppDataDirectory : string;

```

```

121: procedure SetAppDataDirectory(Dir : string);
122: function GetAppSettingsDirectory : string;
123: procedure SetAppSettingsDirectory(Dir : string);
124: function GetAppLogbooksDirectory : string;
125: procedure SetAppLogbooksDirectory(Dir : string);
126: function GetAppBackupsDirectory : string;
127: procedure SetAppBackupsDirectory(Dir : string);
128: //=====
129: // The following Properties are in fact variables that once initialized may be
130: // changed during program execution and are saved in the AppSettings database.
131: //=====
132: function GetAppSettingsInitialPageName : string;
133: procedure SetAppSettingsInitialPageName(PageName : string);
134: function GetAppSettingsInitialPageNum : string;
135: procedure SetAppSettingsInitialPageNum(PageNum : string);
136: function GetAppDatabaseName : string;
137: procedure SetAppDatabaseName(DBName : string);
138:
139: public
140: //=====
141: // The following Properties are in fact constants that once initialized will not be
142: // changed during program execution.
143: //=====
144: property pAppVersion : string
145:         read GetAppVersion
146:         write SetAppVersion;
147: property pAppFullFilePathName : string
148:         read GetAppFullFilePathName
149:         write SetAppFullFilePathName;
150: property pAppFilePath : string read GetAppFilePath
151:         write SetAppFilePath;
152: property pAppFullFileName : string
153:         read GetAppFullFileName
154:         write SetAppFullFileName;
155: property pAppFileName : string
156:         read GetAppFileName
157:         write SetAppFileName;
158: property pAppFileExt : string
159:         read GetAppFileExt;
160: property pAppUserDirectory : string
161:         read GetAppUserDirectory
162:         write SetAppUserDirectory;
163: property pAppDataDirectory : string
164:         read GetAppDataDirectory
165:         write SetAppDataDirectory;
166: property pAppSettingsDirectory : string
167:         read GetAppSettingsDirectory
168:         write SetAppSettingsDirectory;
169: property pAppLogbooksDirectory : string
170:         read GetAppLogbooksDirectory
171:         write SetAppLogbooksDirectory;
172: property pAppBackupsDirectory : string
173:         read GetAppBackupsDirectory
174:         write SetAppBackupsDirectory;
175: property pAppDatabaseName : string
176:         read GetAppDatabaseName
177:         write SetAppDatabaseName;
178:
179: //=====
180: // The following Properties are in fact variables that once initialized may be

```

```

181:      // changed during program execution and are saved in the AppSettings database.
182:      //=====
183:      property pAppSettingsInitialPageName : string
184:          read GetAppSettingsInitialPageName
185:          write SetAppSettingsInitialPageName;
186:      property pAppSettingsInitialPageNum : string
187:          read GetAppSettingsInitialPageNum
188:          write SetAppSettingsInitialPageNum;
189:
190:  end; // TfrmSettings
191:
192: var
193:   frmSettings: TfrmSettings;
194:
195: implementation
196:
197: {$R *.lfm}
198:
199: //=====
200: //          PRIVATE CONSTANTS
201: //=====
202: const
203:
204:   // Directories
205:   cstrAppFullFileName = 'RVMasterLog.exe';
206:   cstrSettingsDirectoryName = 'Settings';
207:   cstrLogbooksDirectoryName = 'Logbooks';
208:   cstrBackupsDirectoryName = 'Backups';
209:   cstrAppDataDirectoryName = 'AppData';
210:
211:   // Databases
212:   cstrApplicationDBName = 'RVMAApplicationDB.sl3';
213:   cstrApplicationSettingsTableName = 'ApplicationSettingsTable';
214:
215:   // Properties
216:   cstrpAppSettingsInitialPageName = 'pAppSettingsInitialPageName';
217:   cstrpAppSettingsInitialPageNum = 'pAppSettingsInitialPageNum';
218:
219:   // frmSettings
220:   straryPageNames : array[0..3] of string = ('Directories', 'Application Settings',
221:                                               'Registration Settings', 'SettingsDB');
222:
223: //=====
224: //          PRIVATE VARIABLES
225: //=====
226: var
227:   vintAppSettingsInitialDirectory : integer;
228:
229: //=====
230: //          PRIVATE ROUTINES
231: //=====
232:
233: //=====
234: //          PUBLIC ROUTINES
235: //=====
236: function TfrmSettings.CreateApplicationDataBase: Boolean;
237: var
238:   vstrTstr : String;
239: begin
240:

```

```

241: showMessage('Creating ApplicationDataBase');
242:
243: DBConnection.Close; // Ensure any connection is closed when we start
244:
245: Result := True;
246:
247: //=====
248: // Create the Default database and tables
249: //=====
250: try
251:
252: //=====
253: // Create the Database
254: //=====
255: DBConnection.Open;
256: DBTransaction.Active := true;
257:
258: //=====
259: // Create the "ApplicationSettingsTable"
260: //=====
261:
262: // showMessage('Create ApplicationSettingsTable');
263:
264: DBConnection.ExecuteDirect('CREATE TABLE "ApplicationSettingsTable"('+
265:                             ' "Property" String PRIMARY KEY, '+
266:                             ' "Value" String );');
267:
268: // showMessage('Create ApplicationSettingsIndex');
269:
270: // Creating an index based upon Property in the ApplicationSettingsTable
271: DBConnection.ExecuteDirect('CREATE UNIQUE INDEX ' +
272:                             ' "ApplicationSettingsTable_Property_idx" ' +
273:                             ' ON "ApplicationSettingsTable" ( "Property" );');
274:
275: //=====
276: // Create the "RegistrationSettingsTable"
277: //=====
278:
279: // showMessage('Create RegistrationSettingsTable');
280:
281: DBConnection.ExecuteDirect('CREATE TABLE "RegistrationSettingsTable"('+
282:                             ' "Property" String PRIMARY KEY, '+
283:                             ' "Value" String );');
284:
285: // showMessage('Create RegistrationSettings Index');
286:
287:
288: DBConnection.ExecuteDirect('CREATE UNIQUE INDEX ' +
289:                             ' "RegistrationSettingsTable_Property_idx" ' +
290:                             ' ON "RegistrationSettingsTable" ( "Property" );');
291: DBTransaction.Commit;
292:
293: //=====
294: // Add the User Adaptble Property Records with Initial Default values.
295: //=====
296:
297: //=====
298: // Application Settings properties
299: //=====
300: DBConnection.ExecuteDirect('INSERT INTO ApplicationSettingsTable VALUES' +

```

```

301:         ' ("pAppSettingsInitialPageName", "Application Settings")')
302:     ;
303:     DBConnection.ExecuteDirect('INSERT INTO ApplicationSettingsTable VALUES' +
304:         ' ("pAppSettingsInitialPageNum", "1")');
305:     //=====
306:     // HURRegistration properties
307:     //=====
308:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
309:         ' ("pRegFirstName", " ")');
310:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
311:         ' ("pRegLastName", " ")');
312:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
313:         ' ("pRegEmailaddress", " ")');
314:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
315:         ' ("pRegCallsign", " ")');
316:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
317:         ' ("pRegKey", " ")');
318:     DBConnection.ExecuteDirect('INSERT INTO RegistrationSettingsTable VALUES' +
319:         ' ("pRegUserID", " ")');
320:     // Additional records go here
321:
322:     //=====
323:     // Commit the additions
324:     //=====
325:
326:     DBTransaction.Commit;
327:
328:     except
329:         ShowMessage('Unable to Create new Database');
330:         Result := False;
331:     end; // Try to Create the Default database and tables
332:
333:     //=====
334:     // Database Created
335:     //=====
336:
337:     DBTransaction.Active := False;
338:     DBConnection.Close;
339:
340:     showmessage('Settings DataBase Created');
341:
342: end; // function TfrmSettings.CreateSettingsDataBase
343:
344: //=====
345: function TfrmSettings.LoadApplicationDatabase : boolean;
346: var
347:     vstrTStr : string;
348:     vintRecNr : integer;
349:
350: begin
351:
352:     showmessage('LoadApplicationDatabase');
353:
354:     Result := True;
355:
356:     try {LoadApplicationDatabase}
357:
358:     //***** showmessage('Opening DBConnection');
359:

```

```

360:         if not DBConnection.Connected then
361:             DBConnection.Open;
362:
363: //         showmessage('DBConnection Open');
364:
365:         if not DBConnection.Connected then
366:         begin
367:             showmessage('Error connecting to the Database. Aborting data loading');
368:             Result := False;
369:             Exit;
370:         end; // if not DBConnection.Connected
371:
372: //=====
373: // Load the Application Settings Table properties
374: //=====
375: sqlqApplicationSettingsTable.SQL.Text :=
376:     'select ' +
377:     ' e.Property, ' +
378:     ' e.Value ' +
379:     'from ApplicationSettingsTable e';
380:
381: DBTransaction.StartTransaction;
382:
383: //         showmessage('DBTransaction.StartTransaction');
384:
385: sqlqApplicationSettingsTable.Open;
386:
387: //         showmessage('sqlqApplicationSettingsTable.Open');
388:
389: // Get pAppSettingsInitialPageName
390: //         showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo    )));
391: pAppSettingsInitialPageName := sqlqApplicationSettingsTable.Fields[1].AsString;
392:
393: // Get pAppSettingsInitialPageNum
394: sqlqApplicationSettingsTable.NEXT;
395: //         showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo    )));
396: pAppSettingsInitialPageNum := sqlqApplicationSettingsTable.Fields[1].AsString;
397: //         showmessage('pAppSettingsInitialPageNum = ' + pAppSettingsInitialPageNum);
398:
399: //=====
400: // Application Settings Table properties  Loaded
401: //=====
402: //         showmessage('ApplicationSettingsTable.EOF');
403: sqlqApplicationSettingsTable.Close;
404:
405:
406: //=====
407: // Load the Registration Table properties
408: //=====
409: sqlqRegistrationSettingsTable.SQL.Text :=
410:     'select ' +
411:     ' e.Property, ' +
412:     ' e.Value ' +
413:     'from RegistrationSettingsTable e';
414:
415: sqlqRegistrationSettingsTable.Open;
416: //         showmessage('sqlqRegistrationSettingsTable Open');
417:
418: // Get pRegFirstName
419: //         showmessage('Record = ' + (IntToStr(sqlqRegistrationSettingsTable.RecNo    )));

```

```

420:     dlgHURegistration.pRegFirstName := sqlqRegistrationSettingsTable.Fields[1].AsString;
421:
422:     // Get pRegLastName
423:     sqlqRegistrationSettingsTable.NEXT;
424:     dlgHURegistration.pRegLastName := sqlqRegistrationSettingsTable.Fields[1].AsString;
425:
426:     // Get pRegEmailAddress
427:     sqlqRegistrationSettingsTable.NEXT;
428:     dlgHURegistration.pRegEmailAddress := sqlqRegistrationSettingsTable.Fields[1].AsString;
429:
430:     // Get pRegCallsign
431:     sqlqRegistrationSettingsTable.NEXT;
432:     dlgHURegistration.pRegCallsign := sqlqRegistrationSettingsTable.Fields[1].AsString;
433:
434:     // Get pRegKey
435:     sqlqRegistrationSettingsTable.NEXT;
436:     dlgHURegistration.pRegKey := sqlqRegistrationSettingsTable.Fields[1].AsString;
437:
438:     // Get pRegUserID
439:     sqlqRegistrationSettingsTable.NEXT;
440:     dlgHURegistration.pRegUserID := sqlqRegistrationSettingsTable.Fields[1].AsString;
441:
442:     //=====
443:     // Registration Table properties  Loaded
444:     //=====
445:     showMessage('RegistrationTable.EOF');
446:     sqlqRegistrationSettingsTable.Close;
447:
448: except
449:
450:     on D: EDatabaseError do
451:     begin
452:         MessageDlg('Error', 'A Database error has occurred. Technical error message: ' +
453:             D.Message, mtError, [mbOK], 0);
454:         Result := False;
455:     end;// on D: EDatabaseError
456:
457: end;// Try {LoadApplicationDatabase}
458:
459: DBTransaction.Active := False;
460: DBConnection.Close;
461:
462: end;// function TfrmSettings.LoadApplicationDatabase
463:
464: //=====
465: function TfrmSettings.SaveApplicationDatabase : Boolean;
466: {
467: var
468:     vstrTStr : string;
469:     vintRecNr : integer;
470: }
471: begin
472: {
473:     showMessage('SaveApplicationDatabase');
474:
475:     Result := True;
476:
477:     try {LoadApplicationDatabase}
478:
479: //*****     showMessage('Opening DBConnection');

```



```
480:
481:     if not DBConnection.Connected then
482:         DBConnection.Open;
483:
484: //     showmessage('DBConnection Open');
485:
486:     if not DBConnection.Connected then
487:     begin
488:         showmessage('Error connecting to the Database. Aborting data loading');
489:         Result := False;
490:         Exit;
491:     end; // if not DBConnection.Connected
492:
493: //=====
494: // Load the Application Settings Table properties
495: //=====
496: sqlqApplicationSettingsTable.SQL.Text :=
497:     'select ' +
498:     ' e.Property, ' +
499:     ' e.Value ' +
500:     'from ApplicationSettingsTable e';
501:
502: DBTransaction.StartTransaction;
503:
504: //     showmessage('DBTransaction.StartTransaction');
505:
506: sqlqApplicationSettingsTable.Open;
507:
508: //     showmessage('sqlqApplicationSettingsTable.Open');
509:
510: // Get pAppSettingsInitialPageName
511: //     showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo    )));
512: pAppSettingsInitialPageName := sqlqApplicationSettingsTable.Fields[1].AsString;
513:
514: // Get pAppSettingsInitialPageNum
515: sqlqApplicationSettingsTable.NEXT;
516: //     showmessage('Record = ' + (IntToStr(sqlqApplicationSettingsTable.RecNo    )));
517: pAppSettingsInitialPageNum := sqlqApplicationSettingsTable.Fields[1].AsString;
518: //     showmessage('pAppSettingsInitialPageNum = ' + pAppSettingsInitialPageNum);
519:
520: //=====
521: // Application Settings Table properties  Loaded
522: //=====
523: //     showmessage('ApplicationSettingsTable.EOF');
524: sqlqApplicationSettingsTable.Close;
525:
526:
527: //=====
528: // Load the Registration Table properties
529: //=====
530: sqlqRegistrationSettingsTable.SQL.Text :=
531:     'select ' +
532:     ' e.Property, ' +
533:     ' e.Value ' +
534:     'from RegistrationSettingsTable e';
535:
536: sqlqRegistrationSettingsTable.Open;
537: //     showmessage('sqlqRegistrationSettingsTable Open');
538:
539: // Get pRegFirstName
```

```

540: //      showMessage('Record = ' + (IntToStr(sqlqRegistrationSettingsTable.RecNo  )));
541:      dlgHURegistration.pRegFirstName := sqlqRegistrationSettingsTable.Fields[1].AsString;
542:
543:      // Get pRegLastName
544:      sqlqRegistrationSettingsTable.NEXT;
545:      dlgHURegistration.pRegLastName := sqlqRegistrationSettingsTable.Fields[1].AsString;
546:
547:      // Get pRegEmailAddress
548:      sqlqRegistrationSettingsTable.NEXT;
549:      dlgHURegistration.pRegEmailAddress := sqlqRegistrationSettingsTable.Fields[1].AsString;
550:
551:      // Get pRegCallsign
552:      sqlqRegistrationSettingsTable.NEXT;
553:      dlgHURegistration.pRegCallsign := sqlqRegistrationSettingsTable.Fields[1].AsString;
554:
555:      // Get pRegKey
556:      sqlqRegistrationSettingsTable.NEXT;
557:      dlgHURegistration.pRegKey := sqlqRegistrationSettingsTable.Fields[1].AsString;
558:
559:      // Get pRegUserID
560:      sqlqRegistrationSettingsTable.NEXT;
561:      dlgHURegistration.pRegUserID := sqlqRegistrationSettingsTable.Fields[1].AsString;
562:
563:      //=====
564:      // Registration Table properties  Loaded
565:      //=====
566:      showMessage('RegistrationTable.EOF');
567:      sqlqRegistrationSettingsTable.Close;
568:
569:  except
570:
571:      on D: EDatabaseError do
572:      begin
573:          MessageDlg('Error', 'A Database error has ocured. Technical error message: ' +
574:              D.Message, mtError, [mbOK], 0);
575:          Result := False;
576:      end;// on D: EDatabaseError
577:
578:  end;// Try {LoadApplicationDatabase}
579:
580:  DBTransaction.Active := False;
581:  DBConnection.Close;
582: }
583: end;// function TfrmSettings.SaveSettingsDataBase
584:
585: //=====
586: //      PROPERTY ROUTINES
587: //=====
588: function TfrmSettings.GetAppVersion: string;
589: begin
590:     Result := fAppVersion;
591: end;// function TfrmSettings.GetVersion
592:
593: //-----
594: procedure TfrmSettings.SetAppVersion(Version: string);
595: begin
596:     fAppVersion := Version;
597: end;// procedure TfrmSettings.SetAppVersion
598:
599: //=====

```

```
600: function TfrmSettings.GetAppFullFilePathName: string;
601: begin
602:     Result := fAppFullFilePathName;
603: end; // function TfrmSettings.GetAppFullFilePathName
604:
605: //-----
606: procedure TfrmSettings.SetAppFullFilePathName(PathName: string);
607: begin
608:     fAppFullFilePathName := PathName;
609: end; // procedure TfrmSettings.SetAppFullFilePathName
610:
611: //=====
612: function TfrmSettings.GetAppFilePath: string;
613: begin
614:     Result := fAppFilePath;
615: end; // function TfrmSettings.GetAppFullFilePath
616:
617: //-----
618: procedure TfrmSettings.SetAppFilePath(Path: string);
619: begin
620:     fAppFilePath := Path;
621: end; // procedure TfrmSettings.SetAppFullFilePath
622:
623: //=====
624: function TfrmSettings.GetAppFullFileName: string;
625: begin
626:     Result := fAppFullFileName;
627: end; // function TfrmSettings.GetAppFullFileNamel
628:
629: //-----
630: procedure TfrmSettings.SetAppFullFileName(FName: string);
631: begin
632:     fAppFullFileName := cstrAppFullFileName;
633: end; // procedure TfrmSettings.SetAppFullFileName
634:
635: //=====
636: function TfrmSettings.GetAppFileName: string;
637: begin
638:     Result := fAppFileName;
639: end; // function TfrmSettings.GetAppFileNamel
640:
641: //-----
642: procedure TfrmSettings.SetAppFileName(FName: string);
643: begin
644:     fAppFileName := FName;
645: end; // procedure TfrmSettings.SetAppFileName
646:
647: //=====
648: function TfrmSettings.GetAppFileExt: string;
649: begin
650:     Result := fAppFileExt;
651: end; // function TfrmSettings.GetAppFileExt
652:
653: //=====
654: function TfrmSettings.GetAppUserDirectory: string;
655: begin
656:     Result := fAppUserDirectory;
657: end; // procedure TfrmSettings.GetAppUserDirectory
658:
659: //-----
```

```
660: procedure TfrmSettings.SetAppUserDirectory(Dir: string);
661: begin
662:     fAppUserDirectory := Dir;
663: end; // procedure TfrmSettings.SetAppUserDataDirectory
664:
665: //=====
666: function TfrmSettings.GetAppDataDirectory: string;
667: begin
668:     Result := fAppDataDirectory;
669: end; // procedure TfrmSettings.GetAppDirectory
670:
671: //-----
672: procedure TfrmSettings.SetAppDataDirectory(Dir: string);
673: begin
674:     fAppDataDirectory := Dir;
675: end; // procedure TfrmSettings.SetAppDataDirectory
676:
677: //=====
678: function TfrmSettings.GetAppSettingsDirectory: string;
679: begin
680:     Result := fAppSettingsDirectory;
681: end; // procedure TfrmSettings.GetAppSettingsDirectory
682:
683: //-----
684: procedure TfrmSettings.SetAppSettingsDirectory(Dir: string);
685: begin
686:     fAppSettingsDirectory := Dir;
687: end; // procedure TfrmSettings.SetAppSettingsDirectory
688:
689: //=====
690: function TfrmSettings.GetAppLogbooksDirectory: string;
691: begin
692:     Result := fAppLogbooksDirectory;
693: end; // procedure TfrmSettings.GetAppLogbooksDirectory
694:
695: //-----
696: procedure TfrmSettings.SetAppLogbooksDirectory(Dir: string);
697: begin
698:     fAppLogbooksDirectory := Dir;
699: end; // procedure TfrmSettings.SeApptLogbooksDirectory
700:
701: //=====
702: function TfrmSettings.GetAppBackupsDirectory: string;
703: begin
704:     Result := fAppBackupsDirectory;
705: end; // procedure TfrmSettings.GetAppBackupsDirectory
706:
707: //-----
708: procedure TfrmSettings.SetAppBackupsDirectory(Dir: string);
709: begin
710:     fAppBackupsDirectory := Dir;
711: end; // procedure TfrmSettings.SetAppBackupsDirectory
712:
713: //=====
714: function TfrmSettings.GetAppSettingsInitialPageName: string;
715: begin
716:     Result := fAppSettingsInitialPageName;
717: end; // procedure TfrmSettings.GetAppSettingsInitialPageName
718:
719: //-----
```

```
720: procedure TfrmSettings.SetAppSettingsInitialPageName(PageName: string);
721: begin
722:     fAppSettingsInitialPageName := PageName;
723: end; // procedure TfrmSettings.SetAppSettingsInitialPageName
724:
725: //=====
726: function TfrmSettings.GetAppSettingsInitialPageNum: string;
727: begin
728:     Result := fAppSettingsInitialPageNum ;
729: end; // procedure TfrmSettings.GetAppSettingsInitialPageNum
730:
731: //-----
732: procedure TfrmSettings.SetAppSettingsInitialPageNum(PageNum: string);
733: begin
734:     fAppSettingsInitialPageNum := PageNum;
735: end; // procedure TfrmSettings.SetAppSettingsInitialPageNum
736:
737: //=====
738: function TfrmSettings.GetAppDatabaseName: string;
739: begin
740:     Result := fAppDatabaseName;
741: end; // procedure TfrmSettings.GetAppDatabaseName
742:
743: //-----
744: procedure TfrmSettings.SetAppDatabaseName(DBName: string);
745: begin
746:     fAppDatabaseName := DBName;
747: end; // procedure TfrmSettings.SetAppDatabaseName
748:
749: { //=====
750: function TfrmSettings.GetAppOwnerFirstName: string;
751: begin
752:     Result := fAppOwnerFirstName;
753: end; // procedure TfrmSettings.GetAppOwnerFirstName
754:
755: //-----
756: procedure TfrmSettings.SetAppOwnerFirstName(FirstName: string);
757: begin
758:     fAppOwnerFirstName := FirstName;
759: end; // procedure TfrmSettings.SetAppOwnerFirstName
760:
761: //=====
762: function TfrmSettings.GetAppOwnerLastName: string;
763: begin
764:     Result := fAppOwnerLastName;
765: end; // procedure TfrmSettings.GetAppOwnerLastName
766:
767: //-----
768: procedure TfrmSettings.SetAppOwnerLastName(LastName: string);
769: begin
770:     fAppOwnerLastName := LastName;
771: end; // procedure TfrmSettings.SetAppOwnerLastName
772:
773: //=====
774: function TfrmSettings.GetAppOwnerCallsign: string;
775: begin
776:     Result := fAppOwnerCallsign;
777: end; // procedure TfrmSettings.GetAppOwnerCallsign
778:
779: //-----
```

```
780: procedure TfrmSettings.SetAppOwnerCallsign(Callsign: string);
781: begin
782:     fAppOwnerCallsign := Callsign;
783: end; // procedure TfrmSettings.SetAppOwnerCallsign
784:
785: //=====
786: function TfrmSettings.GetAppOwnerEmailAddress: string;
787: begin
788:     Result := fAppOwnerEmailAddress;
789: end; // procedure TfrmSettings.GetAppOwnerEmailAddress
790:
791: //-----
792: procedure TfrmSettings.SetAppOwnerEmailAddress(EmailAddress: string);
793: begin
794:     fAppOwnerEmailAddress := EMailAddress;
795: end; // procedure TfrmSettings.SetAppOwnerEmailAddress
796:
797: //=====
798: function TfrmSettings.GetAppOwnerID: string;
799: begin
800:     Result := fAppOwnerID;
801: end; // procedure TfrmSettings.GetAppUserID
802:
803: //-----
804: procedure TfrmSettings.SetAppOwnerID(OwnerID: string);
805: begin
806:     fAppOwnerID := OwnerID;
807: end; // procedure TfrmSettings.SetAppOwnerID }
808:
809: //=====
810: //          MENU ROUTINES
811: //=====
812:
813: //=====
814: //          COMMAND BUTTON ROUTINES
815: //=====
816: procedure TfrmSettings.bbtCancelCloseClick(Sender: TObject);
817: begin
818:
819: end; // procedure TfrmSettings.bbtCancelClick
820:
821: //-----
822: procedure TfrmSettings.bbtOkCloseClick(Sender: TObject);
823: begin
824:
825: end; // procedure TfrmSettings.bbtOkClick
826:
827: //=====
828: //          CONTROL ROUTINES
829: //=====
830:
831: //=====
832: //          FILE ROUTINES
833: //=====
834:
835: //=====
836: //          FORM ROUTINES
837: //=====
838: procedure TfrmSettings.FormCreate(Sender: TObject);
839: begin
```

```

840:
841: //=====
842: // Initialize the Application Directories
843: //=====
844:
845: pAppFileName := Copy (cstrAppFullFileName, 1, 12);
846: pAppFilePath := GetCurrentDir;
847: pAppUserDirectory := (GetWindowsSpecialDir(CSIDL_PERSONAL)) + pAppFileName;
848:
849: // The following directories are created by the INNO Setup Script
850: pAppSettingsDirectory := pAppUserDirectory + '\' + cstrSettingsDirectoryName;
851: pAppLogbooksDirectory := pAppUserDirectory + '\' + cstrLogbooksDirectoryName;
852: pAppBackupsDirectory := pAppUserDirectory + '\' + cstrBackupsDirectoryName;
853: pAppDataDirectory := pAppUserDirectory + '\' + cstrAppDataDirectoryName;
854: pAppDatabaseName := pAppDataDirectory + '\' + cstrApplicationDBName;
855:
856: //=====
857: // Initialize the Data Components
858: //=====
859: DBConnection.DatabaseName := pAppDatabaseName;
860: DBConnection.Transaction := DBTransaction;
861: sqlqApplicationSettingsTable.DataBase := DBConnection;
862: sqlqApplicationSettingsTable.DataSource := dsApplicationSettingsTable;
863: sqlqRegistrationSettingsTable.DataBase := DBConnection;
864: sqlqRegistrationSettingsTable.DataSource := dsRegistrationSettingsTable;
865:
866: //=====
867: // Initialize the Controls
868: //=====
869:
870: // PageControl properties
871: pgDirectories.Caption:=straryPageNames[0];
872: pgApplicationSettings.Caption:=straryPageNames[1];
873: pgRegistrationSettings.Caption:=straryPageNames[2];
874: pgSettingsDB.Caption:=straryPageNames[3];
875:
876: // strgApplicationSettings properties
877: strgrdApplicationSettings.AutoFillColumns := True;
878: strgrdApplicationSettings.AutoSizeColumns;
879:
880: // strgRegistrationSettings properties
881: strgrdRegistrationSettings.AutoFillColumns := True;
882: strgrdRegistrationSettings.AutoSizeColumns;
883:
884: end; // procedure TfrmSettings.FormCreate
885:
886: //-----
887: procedure TfrmSettings.FormShow(Sender: TObject);
888: var
889:   vstrTstr : string;
890:   vintTInt : Integer;
891: begin
892:
893: //=====
894: // initialize form controls
895: //=====
896:
897: // Command Buttons
898:
899: // Directories Page

```

```
900:   edtApplicationDirectory.Text:= pAppFilePath;
901:   edtSettingsDirectory.Text:=pAppSettingsDirectory;
902:   edtLogbooksDirectory.Text:=pAppLogbooksDirectory;
903:   edtBackupsDirectory.Text := pAppBackupsDirectory;
904:   edtAppDataDirectory.Text := pAppDataDirectory;
905:
906:   // Application Settings Page
907:
908:   // strgApplicationSettings
909:   strgrdApplicationSettings.Cells[0,1] := 'pAppSettingsInitialPageNum';
910:   strgrdApplicationSettings.Cells[1,1] := pAppSettingsInitialPageNum;
911:   strgrdApplicationSettings.Cells[0,2] := 'pAppSettingsInitialPageName';
912:   strgrdApplicationSettings.Cells[1,2] := pAppSettingsInitialPageName;
913:
914:   // strgRegistrationSettings
915:   strgrdRegistrationSettings.Cells[0,1] := 'dlgHURegistration.pRegFirstName';
916:   strgrdRegistrationSettings.Cells[1,1] := dlgHURegistration.pRegFirstName;
917:   strgrdRegistrationSettings.Cells[0,2] := 'dlgHURegistration.pRegLastName';
918:   strgrdRegistrationSettings.Cells[1,2] := dlgHURegistration.pRegLastName;
919:   strgrdRegistrationSettings.Cells[0,3] := 'dlgHURegistration.pRegEmailAddress';
920:   strgrdRegistrationSettings.Cells[1,3] := dlgHURegistration.pRegEmailAddress;
921:   strgrdRegistrationSettings.Cells[0,4] := 'dlgHURegistration.pRegCallsign';
922:   strgrdRegistrationSettings.Cells[1,4] := dlgHURegistration.pRegCallsign;
923:   strgrdRegistrationSettings.Cells[0,5] := 'dlgHURegistration.pRegKey';
924:   strgrdRegistrationSettings.Cells[1,5] := dlgHURegistration.pRegKey;
925:   strgrdRegistrationSettings.Cells[0,6] := 'dlgHURegistration.pRegUserID';
926:   strgrdRegistrationSettings.Cells[1,6] := dlgHURegistration.pRegUserID;
927:
928: end;// procedure TfrmAppSetup.FormShow
929:
930: //-----
931: procedure TfrmSettings.FormClose(Sender: TObject; var CloseAction: TCloseAction);
932: begin
933:
934:
935:
936:
937:
938:
939:   DBConnection.Connected := False;
940: end;// procedure TfrmAppSetup.FormClose
941:
942: //=====
943: end.// unit AppSettings
944:
```