

```

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: //
15: // Calls :
16: //
17: // Ver. : 1.0.0
18: //
19: // Date : 19 Apr 2019
20: //
21: //=====
22:
23: interface
24:
25: uses
26:   Buttons, Classes, ComCtrls, Controls, Dialogs, FileUtil, Forms, Graphics, INIFiles,
27:   StdCtrls, SysUtils,
28:   //App Units
29:   // HULib units
30:   HUMessageBoxes;
31:
32: type
33:
34:   { TfrmSettings }
35:
36:   TfrmSettings = class(TForm)
37:     bbtCancel: TBitBtn;
38:     bbtOk: TBitBtn;
39:     edtBackupsDirectory: TEdit;
40:     edtLogbooksDirectory: TEdit;
41:     edtSettingsDirectory: TEdit;
42:     edtApplicationDirectory: TEdit;
43:     Label1: TLabel;
44:     Label2: TLabel;
45:     Label3: TLabel;
46:     Label4: TLabel;
47:     Label5: TLabel;
48:     pcSettings: TPageControl;
49:     pgDirectories: TTabSheet;
50:     procedure bbtCancelClick(Sender: TObject);
51:     procedure bbtOkClick(Sender: TObject);
52:     procedure edtSettingsDirectoryMouseUp(Sender: TObject;
53:       Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
54:     procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);
55:     procedure FormCreate(Sender: TObject);
56:     procedure FormShow(Sender: TObject);
57:
58:   private
59:     fApplicationDirectory : string;
60:     fAppName : string;

```

```

61: fSystemUserDirectory : string;
62: fUserDataDirectory : string;
63: fSettingsDirectory : string;
64: fLogbooksDirectory : string;
65: fBackupsDirectory : string;
66: fOwnerFirstName : string;
67: fOwnerLastName : string;
68: fOwnerCallsign : string;
69: fOwnerEmailAddress : string;
70: fOwnerID : string;
71: function GetApplicationDirectory : string;
72: procedure SetApplicationDirectory(Dir : string);
73: function GetAppName : string;
74: procedure SetAppName(AppName : string);
75: function GetSystemUserDirectory : string;
76: procedure SetSystemUserDirectory(Dir : string);
77: function GetUserDataDirectory : string;
78: procedure SetUserDataDirectory(Dir : string);
79: function GetSettingsDirectory : string;
80: procedure SetSettingsDirectory(Dir : string);
81: function GetLogbooksDirectory : string;
82: procedure SetLogbooksDirectory(Dir : string);
83: function GetBackupsDirectory : string;
84: procedure SetBackupsDirectory(Dir : string);
85: function GetOwnerFirstName : string;
86: procedure SetOwnerFirstName (FirstName : string);
87: function GetOwnerLastName : string;
88: procedure SetOwnerLastName (LastName : string);
89: function GetOwnerCallsign : string;
90: procedure SetOwnerCallsign (Callsign : string);
91: function GetOwnerEmailAddress : string;
92: procedure SetOwnerEmailAddress (EmailAddress : string);
93: function GetOwnerID : string;
94: procedure SetOwnerID (OwnerID : string);
95:
96: public
97:
98:     property pApplicationDirectory : string read GetApplicationDirectory write
SetApplicationDirectory;
99:     property pAppName : string read GetAppName write SetAppName;
100:    property pSystemUserDirectory : string read GetSystemUserDirectory write
SetSystemUserDirectory;
101:    property pUserDataDirectory : string read GetUserDataDirectory write SetUserDataDirectory;
102:    property pSettingsDirectory : string read GetSettingsDirectory write SetSettingsDirectory;
103:    property pLogbooksDirectory : string read GetLogbooksDirectory write SetLogbooksDirectory;
104:    property pBackupsDirectory : string read GetBackupsDirectory write SetBackupsDirectory;
105:    property pOwnerFirstName : string read GetOwnerFirstName write SetOwnerFirstName;
106:    property pOwnerLastName : string read GetOwnerLastName write SetOwnerLastName;
107:    property pOwnerCallsign : string read GetOwnerCallsign write SetOwnerCallsign;
108:    property pOwnerEmailAddress : string read GetOwnerEmailAddress write SetOwnerEmailAddress;
109:    property pOwnerID : string read GetOwnerID write SetOwnerID;
110:
111:    function UserDataDirectoriesExist : Boolean;
112:    procedure ReadSettingsINIFile;
113:    procedure WriteSettingsINIFile;
114:    function CreateUserDataDirectories : Boolean;
115:
116: end; // TfrmSettings
117:
118: var

```

```
119: frmSettings: TfrmSettings;
120:
121: implementation
122:
123: {$R *.lfm}
124:
125: uses
126:   Main;
127:
128: //=====
129: //          PRIVATE CONSTANTS
130: //=====
131: const
132:
133:   cstrAppName = 'RVMasterLog';
134:   cstrSettingsDirectoryName = 'Settings';
135:   cstrLogbooksDirectoryName = 'Logbooks';
136:   cstrBackupsDirectoryName = 'Backups';
137:   cstrUserDataDirectoryPath = 'AppData\Roaming\RVMasterLog';
138:
139: //=====
140: //          PRIVATE VARIABLES
141: //=====
142:
143: //=====
144: //          PRIVATE ROUTINES
145: //=====
146: function TfrmSettings.UserDataDirectoriesExist : Boolean;
147: begin
148:
149:   if not DirectoryExists(pUserDataDirectory) then
150:   begin
151:
152:     if HUErrorMsgYN('No User Data Directory', 'Initial Installation?') = mrYes then
153:     begin
154:       HUInformationMsgOK('Caption', 'Creating Data Directories');
155:       CreateUserDataDirectories;
156:     end; // if HUErrorMsgYN(' No User Directory', 'Create One ?') = mrYes
157:
158:   end; // if UserDataDirectoriesExist(pUserDirectory)
159:
160: end; // function TfrmAppSetupApplicationDirectoryp.UserFilesExist
161:
162: //=====
163: function TfrmSettings.CreateUserDataDirectories : Boolean;
164: begin
165:
166:   // USER DATA DIRECTORY
167:   pUserDataDirectory := frmSettings.pSystemUserDirectory + cstrUserDataDirectoryPath;
168:
169:   if not CreateDir(pUserDataDirectory) then
170:   begin
171:     showmessage('USER DATA DIRECTORY FAILED');
172:     Result := False;
173:     Main.TerminateApp;
174:   end; // if not CreateDir(pUserDataDirectory)
175:
176:   // SETTINGS DIRECTORY
177:
178:   pSettingsDirectory := pUserDataDirectory + '\' + cstrSettingsDirectoryName;
```

```

179:   if not CreateDir(pSettingsDirectory) then
180:   begin
181:       showmessage('SETTINGS DIRECTORY FAILED');
182:       Result := False;
183:       Main.TerminateApp;
184:   end; // if not CreateDir(pSettingsDirectory)
185:
186:   // LOGBOOKS DIRECTORY
187:   pLogbooksDirectory := pUserDataDirectory + '\' + cstrLogbooksDirectoryName;
188:   if not CreateDir(pLogbooksDirectory) then
189:   begin
190:       showmessage('LOGBOOKS DIRECTORY FAILED');
191:       Result := False;
192:       Main.TerminateApp;
193:   end; // if not CreateDir(pLogbooksDirectory)
194:
195:   // BACKUPS DIRECTORY
196:   pBackupsDirectory := pUserDataDirectory + '\' + cstrBackupsDirectoryName;
197:   if not CreateDir(pBackupsDirectory) then
198:   begin
199:       showmessage('BACKUP DIRECTORY FAILED');
200:       Result := False;
201:       Main.TerminateApp;
202:   end; // if not CreateDir(pBackupsDirectory)
203:
204:   // Load the databases
205:   CopyFile (frmSettings.pApplicationDirectory +
206:       '\ ' + 'UserData' + '\ ' + 'ApplicationDB.sl3',
207:       frmSettings.pUserDataDirectory + '\ ' + 'ApplicationDB.sl3');
208:
209:   CopyFile (frmSettings.pApplicationDirectory +
210:       '\ ' + 'UserData' + '\ ' + 'LogbooksDB.sl3',
211:       frmSettings.pUserDataDirectory + '\ ' + 'LogbooksDB.sl3');
212:
213:   CopyFile (frmSettings.pApplicationDirectory +
214:       '\ ' + 'UserData' + '\ ' + 'ManufacturersDB.sl3',
215:       frmSettings.pUserDataDirectory + '\ ' + 'ManufacturersDB.sl3');
216:
217:   //      CopyFile (frmSettings.pApplicationDirectory +
218:   //          '\ ' + 'UserData' + '\ ' + frmHUGeoDB.pHUGeoDBName,
219:   //          frmHUGeoDB.pHUGeoDBPath);
220:
221:   Result := True;
222:
223: end; // function CreateUserDataDirectories
224:
225: //=====
226: //      PUBLIC ROUTINES
227: //=====
228:
229: //=====
230: //      PROPERTY ROUTINES
231: //=====
232: function TfrmSettings.GetApplicationDirectory: string;
233: begin
234:     Result := fApplicationDirectory;
235: end; // function TfrmSettings.GetApplicationDirectory
236:
237: //-----
238: procedure TfrmSettings.SetApplicationDirectory(Dir: string);

```

```
239: begin
240:     fApplicationDirectory := Dir;
241: end; // procedure TfrmSettings.SetApplicationDirectory
242:
243: //=====
244: function TfrmSettings.GetAppName: string;
245: begin
246:     Result := fAppName;
247: end; // function TfrmSettings.GetAppName
248:
249: //-----
250: procedure TfrmSettings.SetAppName(AppName: string);
251: begin
252:     fAppName := AppName;
253: end; // procedure TfrmSettings.SetAppName
254:
255: //=====
256: function TfrmSettings.GetUserDataDirectory: string;
257: begin
258:     Result := fUserDataDirectory;
259: end; // function TfrmSettings.GetUserDataDirectory
260:
261: //-----
262: procedure TfrmSettings.SetUserDataDirectory(Dir: string);
263: begin
264:     fUserDataDirectory := Dir;
265: end; // procedure TfrmSettings.SetUserDataDirectory
266:
267: //=====
268: function TfrmSettings.GetSystemUserDirectory: string;
269: begin
270:     Result := fSystemUserDirectory;
271: end; // function TfrmSettings.GetSystemUserDirectory
272:
273: //-----
274: procedure TfrmSettings.SetSystemUserDirectory(Dir: string);
275: begin
276:     fSystemUserDirectory := Dir;
277: end; // procedure TfrmSettings.SetSystemUserDirectory
278:
279: //=====
280: function TfrmSettings.GetSettingsDirectory: string;
281: begin
282:     Result := fSettingsDirectory;
283: end; // procedure TfrmSettings.GetAppSettingsDirectory
284:
285: //-----
286: procedure TfrmSettings.SetSettingsDirectory(Dir: string);
287: begin
288:     fSettingsDirectory := Dir;
289: end; // procedure TfrmSettings.SetAppSettingsDirectory
290:
291: //=====
292: function TfrmSettings.GetLogbooksDirectory: string;
293: begin
294:     Result := fLogbooksDirectory;
295: end; // procedure TfrmSettings.GetLogbooksDirectory
296:
297: //-----
298: procedure TfrmSettings.SetLogbooksDirectory(Dir: string);
```

```
299: begin
300:     fLogbooksDirectory := Dir;
301: end;// procedure TfrmSettings.SetLogbooksDirectory
302:
303: //=====
304: function TfrmSettings.GetBackupsDirectory: string;
305: begin
306:     Result := fBackupsDirectory;
307: end;// procedure TfrmSettings.GetBackupsDirectory
308:
309: //-----
310: procedure TfrmSettings.SetBackupsDirectory(Dir: string);
311: begin
312:     fBackupsDirectory := Dir;
313: end;// procedure TfrmSettings.SetBackupsDirectory
314:
315: //=====
316: function TfrmSettings.GetOwnerFirstName: string;
317: begin
318:     Result := fOwnerFirstName;
319: end;// procedure TfrmSettings.GetOwnerFirstName
320:
321: //-----
322: procedure TfrmSettings.SetOwnerFirstName(FirstName: string);
323: begin
324:     fOwnerFirstName := FirstName;
325: end;// procedure TfrmSettings.SetOwnerFirstName
326:
327: //=====
328: function TfrmSettings.GetOwnerLastName: string;
329: begin
330:     Result := fOwnerLastName;
331: end;// procedure TfrmSettings.GetOwnerLastName
332:
333: //-----
334: procedure TfrmSettings.SetOwnerLastName(LastName: string);
335: begin
336:     fOwnerLastName := LastName;
337: end;// procedure TfrmSettings.SetOwnerLastName
338:
339: //=====
340: function TfrmSettings.GetOwnerCallsign: string;
341: begin
342:     Result := fOwnerCallsign;
343: end;// procedure TfrmSettings.GetOwnerCallsign
344:
345: //-----
346: procedure TfrmSettings.SetOwnerCallsign(Callsign: string);
347: begin
348:     fOwnerCallsign := Callsign;
349: end;// procedure TfrmSettings.SetOwnerCallsign
350:
351: //=====
352: function TfrmSettings.GetOwnerEmailAddress: string;
353: begin
354:     Result := fOwnerEmailAddress;
355: end;// procedure TfrmSettings.GetOwnerEmailAddress
356:
357: //-----
358: procedure TfrmSettings.SetOwnerEmailAddress(EmailAddress: string);
```

```
359: begin
360:     fOwnerEmailAddress := EMailAddress;
361: end; // procedure TfrmSettings.SetOwnerEmailAddress
362:
363: //=====
364: function TfrmSettings.GetOwnerID: string;
365: begin
366:     Result := fOwnerID;
367: end; // procedure TfrmSettings.GetUserID
368:
369: //-----
370: procedure TfrmSettings.SetOwnerID(OwnerID: string);
371: begin
372:     fOwnerID := OwnerID;
373: end; // procedure TfrmSettings.SetOwnerID
374:
375: //=====
376: //          MENU ROUTINES
377: //=====
378:
379: //=====
380: //          COMMAND BUTTON ROUTINES
381: //=====
382: procedure TfrmSettings.bbtCancelClick(Sender: TObject);
383: begin
384:
385: end; // procedure TfrmSettings.bbtCancelClick
386:
387: //-----
388: procedure TfrmSettings.bbtOkClick(Sender: TObject);
389: begin
390:
391: end; // procedure TfrmSettings.bbtOkClick
392:
393: //=====
394: //          CONTROL ROUTINES
395: //=====
396: procedure TfrmSettings.edtSettingsDirectoryMouseUp(Sender: TObject;
397:     Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
398: begin
399:     If Button = mbRight then
400:         showmessage('Mouse Up');
401: end; // procedure TfrmSettings.edtSettingsDirectoryMouseUp
402:
403: //=====
404: //          FILE ROUTINES
405: //=====
406: const
407:     cstrApplicationINIFileName = 'RVMasterLog.ini';
408:
409:     cstrUserDirectories = 'USER DIRECTORIES';
410:     cstrKeyuserDirectory = 'User Directory';
411:     cstrKeySettingsDirectory = 'Settings Directory';
412:     cstrKeyLogbooksDirectory = 'Logbooks Directory';
413:     cstrKeyBackupsDirectory = 'Backups Directory';
414:     cstrRegistrationData = 'REGISTRATION DATA';
415:     cstrKeyOwnerFirstName = 'Owner First Name';
416:     cstrKeyOwnerLastName = 'Owner Last Name';
417:     cstrKeyOwnerCallsign = 'Owner Callsign';
418:     cstrKeyOwnerEmailAddress = 'Owner Email Address';
```

[illegible]


```
479:                                     pOwnerLastName);
480:
481:     // Owner Callsign
482: vstrTStr := ApplicationINIFile.ReadString(cstrRegistrationData,
483:                                     cstrKeyOwnerCallsign,
484:                                     pOwnerCallsign);
485:
486:     // Owner Email Address
487: vstrTStr := ApplicationINIFile.ReadString(cstrRegistrationData,
488:                                     cstrKeyOwnerEmailAddress,
489:                                     pOwnerEmailAddress);
490:
491:     // OwnerID
492: vstrTStr := ApplicationINIFile.ReadString(cstrRegistrationData,
493:                                     cstrKeyOwnerID,
494:                                     pOwnerID);
495:
496: ApplicationINIFile.Free;
497:
498: end; // procedure TfrmSettings.ReadSettingsINIFile
499:
500: //-----
501: procedure TfrmSettings.WriteSettingsINIFile;
502: begin
503:
504:     ApplicationINIFileName := pApplicationDirectory + '\' + cstrApplicationINIFileName;
505:     ApplicationINIFile := TINIFile.Create(ApplicationINIFileName);
506:
507:     // DIRECTORY SECTION
508:
509:     // USER Directory
510: ApplicationINIFile.WriteString(cstrUserDirectories,
511:                               cstrKeyUserDirectory,
512:                               pUserDataDirectory);
513:
514: ApplicationINIFile.WriteString(cstrUserDirectories,
515:                               cstrKeySettingsDirectory,
516:                               pSettingsDirectory);
517:
518: ApplicationINIFile.WriteString(cstrUserDirectories,
519:                               cstrKeyLogbooksDirectory,
520:                               pLogbooksDirectory);
521:
522: ApplicationINIFile.WriteString(cstrUserDirectories,
523:                               cstrKeyBackupsDirectory,
524:                               pBackupsDirectory);
525:
526:     // REGISTRATION DATA
527:
528:     // Owner First Name
529: ApplicationINIFile.WriteString(cstrRegistrationData,
530:                               cstrKeyOwnerFirstName,
531:                               pOwnerFirstName);
532:
533:     // Owner Last Name
534: ApplicationINIFile.WriteString(cstrRegistrationData,
535:                               cstrKeyOwnerLastName,
536:                               pOwnerLastName);
537:
538:     // Owner Callsign
```

```

539:     ApplicationINIFile.WriteString(cstrRegistrationData,
540:                                   cstrKeyOwnerCallsign,
541:                                   pOwnerCallsign);
542:
543:     // Owner Email Address
544:     ApplicationINIFile.WriteString(cstrRegistrationData,
545:                                   cstrKeyOwnerEmailAddress,
546:                                   pOwnerEmailAddress);
547:
548:     // OwnerID
549:     ApplicationINIFile.WriteString(cstrRegistrationData,
550:                                   cstrKeyOwnerID,
551:                                   pOwnerID);
552:
553:     ApplicationINIFile.Free;
554:
555: end; // procedure TfrmSettings.WriteSettingsINIFile
556:
557: //=====
558: //          FORM ROUTINES
559: //=====
560: procedure TfrmSettings.FormCreate(Sender: TObject);
561: begin
562:
563:     pAppName := cstrAppName;
564:     pApplicationDirectory := GetCurrentDir;
565:     pSystemUserDirectory := GetUserDir;
566:     pUserDataDirectory := frmSettings.pSystemUserDirectory + cstrUserDataDirectoryPath;
567:     pSettingsDirectory := pUserDataDirectory + '\\' + cstrSettingsDirectoryName;
568:
569: end; // procedure TfrmAppSetup.FormCreate
570:
571: //-----
572: procedure TfrmSettings.FormShow(Sender: TObject);
573: begin
574:
575:     // Load current properties
576:     edtApplicationDirectory.Text:= pApplicationDirectory;
577:     edtSettingsDirectory.Text:=pSettingsDirectory;
578:     edtLogbooksDirectory.Text:=pLogbooksDirectory;
579:     edtBackupsDirectory.Text := pBackupsDirectory;
580:
581: end; // procedure TfrmAppSetup.FormShow
582:
583: //-----
584: procedure TfrmSettings.FormClose(Sender: TObject; var CloseAction: TCloseAction);
585: begin
586:
587: end; // procedure TfrmAppSetup.FormClose
588:
589: //=====
590:
591: end. // unit AppSettings
592:

```