

```

1: unit DataEntry;
2:
3: {$mode objfpc}{$H+}
4:
5: //=====
6: //
7: //  DataEntry.pas
8: //
9: //  Calls: AppTypes
10: //          Constants
11: //          DataEntry_FAV : DataEntry_FAV_Init
12: //                          DataEntry_Fave_Save
13: //          DataEntry_UHFMEM : DataEntry_UHFMEM_Init
14: //                          DataEntry_UHFMEM_Save
15: //          DataEntry_VHFMEM : DataEntry_VHFMEM_Init
16: //                          DataEntry_VHFMEM_Save
17: //          Main
18: //          Utilities : GetToneNrFromIndex
19: //                      GetToneIndexFromToneNr
20: //                      ValidVHFFrequency
21: //
22: //          Variables
23: //
24: //  Called By: DataEntry_FAV : DataEntry_FAV_Init
25: //                          DataEntry_FAV_Save
26: //          DataEntry_MEM : DataEntry_MEM_Init
27: //          Main : ProcessFavButton
28: //          Mem : TfrmMEM.bbtAddClick
29: //              TfrmMEM.bbtEditClick
30: //          TMVFiles : WriteTMVFile
31: //
32: //  Ver: 1.0.0
33: //
34: //  Date: 8 Dec 2013
35: //
36: //=====
37:
38: interface
39:
40: uses
41:   Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, ExtCtrls,
42:   StdCtrls, Buttons,
43:   // Application Units
44:   AppConstants, AppTypes, AppVariables, DataEntry_FAV, DataEntry_UHFMEM, DataEntry_VHFMEM,
45:   Utilities;
46:
47: type
48:
49:   TfrmDataEntry = class(TForm)
50:     bbtSave: TBitBtn;
51:     bbtClear: TBitBtn;
52:     bbtReset: TBitBtn;
53:     bbtCancel: TBitBtn;
54:     chkScan: TCheckBox;
55:     chkDTSS: TCheckBox;
56:     cbxTones: TComboBox;
57:     edtComments: TEdit;
58:     edtChannelName: TEdit;
59:     edtSource: TEdit;
60:     edtRXFrequency: TEdit;

```

```

61:     edtTXFrequency: TEdit;
62:     edtDTSSCode: TEdit;
63:     edtOffsetShift: TEdit;
64:     GroupBox1: TGroupBox;
65:     GroupBox2: TGroupBox;
66:     GroupBox3: TGroupBox;
67:     GroupBox4: TGroupBox;
68:     GroupBox5: TGroupBox;
69:     Label1: TLabel;
70:     Label2: TLabel;
71:     Label3: TLabel;
72:     Label4: TLabel;
73:     Label5: TLabel;
74:     Label6: TLabel;
75:     Label8: TLabel;
76:     rbtRFPowerHigh: TRadioButton;
77:     rbtCTCSS: TRadioButton;
78:     rbtTone: TRadioButton;
79:     rbtNoTones: TRadioButton;
80:     rbtRFPowerLow: TRadioButton;
81:     rbtRFPowerMedium: TRadioButton;
82:     rbtSimplex: TRadioButton;
83:     rbtUHF: TRadioButton;
84:     rbtVHF: TRadioButton;
85:     rbtMinus: TRadioButton;
86:     rbtPlus: TRadioButton;
87:     procedure bbtCancelClick(Sender: TObject);
88:     procedure bbtClearClick(Sender: TObject);
89:     procedure bbtResetClick(Sender: TObject);
90:     procedure bbtSaveClick(Sender: TObject);
91:     procedure chkDTSSChange(Sender: TObject);
92:     procedure edtChannelNameKeyPress(Sender: TObject; var Key: char);
93:     procedure edtCommentsKeyPress(Sender: TObject; var Key: char);
94:     procedure edtDTSSCodeKeyPress(Sender: TObject; var Key: char);
95:     procedure edtRXFrequencyExit(Sender: TObject);
96:     procedure edtRXFrequencyKeyPress(Sender: TObject; var Key: char);
97:     procedure FormActivate(Sender: TObject);
98:     procedure rbtMinusChange(Sender: TObject);
99:     procedure rbtNoTonesChange(Sender: TObject);
100:    procedure rbtPlusChange(Sender: TObject);
101:    procedure rbtSimplexChange(Sender: TObject);
102:    procedure rbtVHFChange(Sender: TObject);
103: private
104:     { private declarations }
105: public
106:     { public declarations }
107:     const
108:         cstrFavFormTitle = 'Favourite Button Data Entry';
109:         cstrMemFormTitle = 'Memory Channel Data Entry';
110:
111:     var
112:         vdetDataEntryType : TDataEntryType;
113:         vbytChannelNumber : Byte;
114:
115:     function CalculateTXFrequency : string;
116:     procedure DisableDTSSCode;
117:     procedure EnableDTSSCode;
118:     procedure SetDTSSCode;
119:     procedure SetShiftOffset;
120:

```

```

121:     end;
122:
123: var
124:     frmDataEntry: TfrmDataEntry;
125:
126: implementation
127:
128: {$R *.lfm}
129:
130: uses
131:     Main;
132:
133: const
134:
135:     cstrInvalidChannelNameMsg = '           Invalid Channel Name' +
136:                                #13 +
137:                                'The Channel is mandatory and must contain' +
138:                                #13 +
139:                                '           5 to 15 characters';
140:
141:     cstrInvalidDTSSCodeMsg = '           Invalid DTSS Code' +
142:                              #13 +
143:                              'The DTSS Code must be in the format nnn' +
144:                              #13 +
145:                              '           and between 000 and 999'
146:                              ;
147:     cstrInvalidVHFFrequencyMsg = '           Invalid VHF RX Frequency.' +
148:                                  #13 +
149:                                  'The frequency must be in the format nnn.nnn' +
150:                                  #13 +
151:                                  '           and between 118.000 and 173.999';
152:
153:     cstrInvalidUHFFrequencyMsg = '           Invalid UHF RX Frequency' +
154:                                  #13 +
155:                                  'The frequency must be in the format nnn.nnn' +
156:                                  #13 +
157:                                  '           and between 400.000 and 469.999';
158:
159:     cstrResetMsg = '  Conifirm that you wish to Reset the form.' +
160:                   #13 +
161:                   '  This action will reset all data fields to' +
162:                   #13 +
163:                   'their original values when the form was opened.';
164:
165:     cstrCancelMsg = '  Conifirm that you wish to Cancel this entry.' +
166:                   #13 +
167:                   'This action will simply close the Data Entry form' +
168:                   #13 +
169:                   '    and make no changes to the original data.';
170:
171:     cstrClearMsg = '  Conifirm that you wish to Clear this entry.' +
172:                   #13 +
173:                   'This action will reset all data entry fields' +
174:                   #13 +
175:                   '           to their default valuse.';
176:
177:     cstrContinueMsg = '  Do you want to continue ?';
178:
179: var
180:

```

```

181:   vbLnValidData : Boolean;
182:
183: //=====
184: //      SUPPORT ROUTINES
185: //=====
186: function TfrmDataEntry.CalculateTXFrequency : string;
187:
188: var
189:   vsngTXFrequency : Single;
190:
191: begin
192:
193:   if Length(frmDataEntry.edtRXFrequency.Text) > 0 then
194:   begin
195:
196:     vsngTXFrequency := StrToFloat(frmDataEntry.edtRXFrequency.Text);
197:
198:     // Calculate based on Band (UHF or VHF) and Shift (Simplex, Plus or Minus)
199:     if frmDataEntry.rbtVHF.Checked then
200:     begin
201:       if frmDataEntry.rbtPlus.Checked then
202:         vsngTXFrequency := vsngTXFrequency + StrToFloat(gcstrVHFShiftOffset)
203:       else
204:         vsngTXFrequency := vsngTXFrequency - StrToFloat(gcstrVHFShiftOffset);
205:     end
206:   else
207:   begin
208:     if frmDataEntry.rbtPlus.Checked then
209:       vsngTXFrequency := vsngTXFrequency + StrToFloat(gcstrUHFShiftOffset)
210:     else
211:       vsngTXFrequency := vsngTXFrequency - StrToFloat(gcstrUHFShiftOffset);
212:   end; // if frmDataEntry.rbtVHF.Checked
213:
214:   Result := Format('%-8.3f', [vsngTXFrequency]);
215: end
216: else
217:   Result := ''; // if Length(frmDataEntry.edtRXFrequency.Text > 0
218:
219: end; // function CalculateTXFrequency
220:
221: //-----
222: procedure TfrmDataEntry.SetShiftOffset;
223: begin
224:
225:   // This routine does a number things, all related to a band change from VHF to UHF.
226:
227:   // It first checks to see if there is a Frequency in the RX Frequency edit box.
228:   // If there is none, the rest doesn't matter and we simply adjust the Shift offset
229:   // value accordingly and Exit.
230:   if Length (frmDataEntry.edtRXFrequency.Text) = 0 then
231:   begin
232:
233:     if frmDataEntry.rbtVHF.Checked then
234:     begin
235:       if frmDataEntry.rbtSimplex.Checked then
236:         frmDataEntry.edtOffsetShift.Text := gcstrNoShiftOffset
237:       else
238:         frmDataEntry.edtOffsetShift.Text := gcstrVHFShiftOffset;
239:     end
240:   else

```

```

241:     begin
242:         if frmDataEntry.rbtSimplex.Checked then
243:             frmDataEntry.edtOffsetShift.Text := gcstrNoShiftOffset
244:         else
245:             frmDataEntry.edtOffsetShift.Text := gcstrUHFShiftOffset;
246:     end; // if frmDataEntry.rbtVHF.Checked
247:
248:     Exit;
249:
250: end; // if Length (frmDataEntry.edtRXFrequency.Text) = 0
251:
252: // There is a frequency in the edit box, so we check the RX Frequency against the
253: // new band selection. If the current frequency is out of band, then an error message
254: // is displayed and the user is asked if they wish to continue. If they do, both the
255: // RX and TX Frequency edit boxes are cleared and the Shift offset is applied
256: // according to the Bnad selected. If they do not wish to continue, the Bnad selection
257: // buttons are returned to their original values and no changes are made.
258: if frmDataEntry.rbtVHF.Checked then
259:     begin
260:
261:         if not ValidVHFFrequency ( frmDataEntry.edtRXFrequency.Text ) then
262:             begin
263:                 if MessageDlg ( cstrInvalidVHFFrequencyMsg + #13 + cstrContinueMsg, mtError,
264:                     [mbYes, mbNo], 0) = mrNo then
265:                     begin
266:                         frmDataEntry.rbtUHF.Checked := True;
267:                         Exit;
268:                     end
269:                 else
270:                     begin
271:                         frmDataEntry.edtRXFrequency.Text := '';
272:                         frmDataEntry.edtTXFrequency.Text := '';
273:                     end; // if MessageDlg ( cstrInvalidVHFFrequencyMsg
274:                 end; // if not ValidVHFFrequency
275:
276:             end
277:         else
278:             begin
279:
280:                 if not ValidUHFFrequency ( frmDataEntry.edtRXFrequency.Text ) then
281:                     begin
282:                         if MessageDlg ( cstrInvalidUHFFrequencyMsg + #13 + cstrContinueMsg, mtError,
283:                             [mbYes, mbNo], 0) = mrNo then
284:                             begin
285:                                 frmDataEntry.rbtVHF.Checked := True;
286:                                 Exit;
287:                             end
288:                         else
289:                             begin
290:                                 frmDataEntry.edtRXFrequency.Text := '';
291:                                 frmDataEntry.edtTXFrequency.Text := '';
292:                             end; // if MessageDlg ( cstrInvalidVHFFrequencyMsg
293:                         end; // if not ValidUHFFrequency
294:
295:                     end; // if frmDataEntry.rbtVHF.Checked
296:
297:                 if frmDataEntry.rbtVHF.Checked then
298:                     begin
299:                         if frmDataEntry.rbtSimplex.Checked then
300:                             frmDataEntry.edtOffsetShift.Text := ''

```

```
301:         else
302:             frmDataEntry.edtOffsetShift.Text := gcstrVHFShiftOffset;
303:         end
304:     else
305:     begin
306:         if frmDataEntry.rbtSimplex.Checked then
307:             frmDataEntry.edtOffsetShift.Text := ''
308:         else
309:             frmDataEntry.edtOffsetShift.Text := gcstrUHFShiftOffset;
310:         end; // if frmDataEntry.rbtVHF.Checked
311:
312:         frmDataEntry.edtTXFrequency.Text := frmDataEntry.CalculateTXFrequency;
313:
314:     end; // procedure SetShifOffset;
315:
316: //-----
317: procedure SetToneFreq;
318: begin
319:     if frmDataEntry.rbtNoTones.Checked then
320:         frmDataEntry.cbxTones.Text := ''
321:     else
322:         frmDataEntry.cbxTones.Text := IntToStr (frmDataEntry.cbxTones.ItemIndex); //Items[0];
323: end; // procedure SetToneFreq;
324:
325: //-----
326: procedure TfrmDataEntry.SetDTSSCode;
327: begin
328:
329:     if frmDataEntry.chkDTSS.Checked then
330:         frmDataEntry.edtDTSSCode.Text :=
331:             gvstrFAVChannelDataArray[frmDataEntry.vbytChannelNumber, gcbytDTSSCodeField]
332:     else
333:         frmDataEntry.edtDTSSCode.Text := '';
334:
335: end; // procedure SetDTSSCode;
336:
337: //-----
338: //      DTSS CODE EDIT BOX ROUTINES
339: //-----
340: procedure TfrmDataEntry.EnabledDTSSCode;
341: begin
342:
343:     frmDataEntry.edtDTSSCode.Enabled := True;
344:     frmDataEntry.edtDTSSCode.Color := clWhite;
345:
346: end; // procedure EnabledDTSSCode;
347:
348: //-----
349: procedure TfrmDataEntry.DisabledDTSSCode;
350: begin
351:
352:     frmDataEntry.edtDTSSCode.Enabled := False;
353:     frmDataEntry.edtDTSSCode.Color := clYellow;
354:
355: end; // procedure DisabledDTSSCode;
356:
357: //=====
358: //      FORM ROUTINES
359: //=====
360: procedure TfrmDataEntry.FormActivate(Sender: TObject);
```

```

361:
362: var
363:   vbytTemp : Integer;
364:
365: begin
366:
367:   // Set the Editbox lengths
368:   edtChannelName.MaxLength := gcbytMaxChannelNameLength;
369:   edtComments.MaxLength := gcbytMaxCommentsLength;
370:   edtDTSSCode.MaxLength := gcbytMaxDTSSCodeLength;
371:   edtRXFrequency.MaxLength := gcbytMaxFrequencyLength;
372:   edtTXFrequency.MaxLength := gcbytMaxFrequencyLength;
373:
374:   // Init the DTSS edit box
375:   DisabledDTSSCode;
376:
377:   // Configure the Command Buttons
378:   bbtSave.Enabled := True;
379:   bbtClear.Enabled := True;
380:   bbtReset.Enabled := True;
381:   frmDataEntry.bbtCancel.Enabled := True;
382:
383:   // Set the initial Tab position
384:   edtRXFrequency.SetFocus;
385:
386:   // The remainder of the form initialization depends on the Data Entry type
387:   case vdetDataEntryType of
388:
389:     detFAV : DataEntry_FAV_Init;
390:   //   detVFO : VFOFormInit;
391:     detUHFMEM : DataEntry_UHFMEM_Init;
392:     detVHFMEM : DataEntry_VHFMEM_Init;
393:
394:   end; // case vdetDataEntryType
395:
396: end; // procedure TfrmDataEntry.FormActivate
397:
398: //=====
399: //      BUTTON ROUTINES
400: //=====
401: procedure TfrmDataEntry.bbtCancelClick(Sender: TObject);
402: begin
403:
404:   if MessageDlg('ConfirmReset', cstrCancelMsg, mtConfirmation, [mbYes, mbNo], 0) = mrYes then
405:     ModalResult := mrClose
406:   else
407:     ModalResult := mrNone;
408:
409: end; // procedure TfrmDataEntry.bbtCancel
410:
411: //-----
412: procedure TfrmDataEntry.bbtClearClick(Sender: TObject);
413:
414: var
415:   vbytTemp1 : Byte;
416:   vbytTemp2 : Byte;
417:
418: begin
419:
420:   //=====

```

```

421: // Clear the data elements
422: //=====
423:
424: if MessageDlg('Confirm Clear', cstrClearMsg, mtConfirmation, [mbYes, mbNo], 0) = mrYes then
425: begin
426:
427:     // The remainder of the form initialization depends on the Data Entry type
428:     case vdetDataEntryType of
429:
430:         detFAV : begin
431:             // The answer was Yes so we clear all of the fields of the record for this
button
432:             for vbytTemp1 := 1 to gcbytMaxChannelFieldCount do
433:             begin
434:                 gvstrFAVChannelDataArray[vbytChannelNumber, vbytTemp1] := '';
435:             end;// for vbytTemp1 := 1 to vbytChannelNumber do
436:
437:             // and clear the Favourite button
438:             case vbytChannelNumber of
439:                 1 : frmMain.bbtFav01.Caption := '';
440:                 2 : frmMain.bbtFav02.Caption := '';
441:                 3 : frmMain.bbtFav03.Caption := '';
442:                 4 : frmMain.bbtFav04.Caption := '';
443:                 5 : frmMain.bbtFav05.Caption := '';
444:                 6 : frmMain.bbtFav06.Caption := '';
445:                 7 : frmMain.bbtFav07.Caption := '';
446:                 8 : frmMain.bbtFav08.Caption := '';
447:                 9 : frmMain.bbtFav09.Caption := '';
448:                 10 : frmMain.bbtFav10.Caption := '';
449:                 11 : frmMain.bbtFav11.Caption := '';
450:                 12 : frmMain.bbtFav12.Caption := '';
451:             end;// case vbytChannelNumber of
452:
453:             end;
454:
455: //         detVFO : VFOFormInit;
456:         detUHFMEM : begin
457:             // The answer was Yes so we clear all of the fields of the record for this
button
458:             for vbytTemp1 := 1 to gcbytMaxChannelFieldCount do
459:             begin
460:                 gvstrUHFChannelDataArray[vbytChannelNumber, vbytTemp1] := '';
461:             end;// for vbytTemp1 := 1 to vbytChannelNumber do
462:             end;
463:
464:         detVHFMEM : begin
465:             // The answer was Yes so we clear all of the fields of the record for this
button
466:             for vbytTemp1 := 1 to gcbytMaxChannelFieldCount do
467:             begin
468:                 gvstrVHFChannelDataArray[vbytChannelNumber, vbytTemp1] := '';
469:             end;// for vbytTemp1 := 1 to vbytChannelNumber do
470:             end;
471:
472:
473:     end;// case vdetDataEntryType of
474:
475:     gvstrTMVDDataChanged := True;
476:
477: end;// if MessageDlg('Confirm Clear'

```



```

478:
479: end;// procedure TfrmDataEntry.bbtClearClick
480:
481: //-----
482: procedure TfrmDataEntry.bbtResetClick(Sender: TObject);
483: begin
484:
485:     // This routine resets the data entry form to the original data that was present when the
486:     // for was first opened. It resets the form data fields to the "Original" data field
487:     // variables.
488:     if MessageDlg('Confirm Reset', cstrResetMsg, mtConfirmation, [mbYes, mbNo], 0) = mrYes then
489:     begin
490:         // The remainder of the form initialization depends on the Data Entry type
491:         case vdetDataEntryType of
492:
493:             detFAV : DataEntry_FAV_Init;
494:         //     detVFO : VFOFormInit;
495:             detUHFMEM : DataEntry_UHFMEM_Init;
496:             detVHFMEM : DataEntry_VHFMEM_Init;
497:
498:         end;// case vdetDataEntryType
499:
500:     end;// if MessageDlg('Confirm Reset',
501:
502:     ModalResult := mrNone;
503:
504: end;// procedure TfrmDataEntry.bbtResetClick
505:
506: //-----
507: procedure TfrmDataEntry.bbtSaveClick(Sender: TObject);
508:
509: var
510:     vstrTStr : string;
511:
512: begin
513:
514:     //=====
515:     // First we validate Data elements
516:     //=====
517:
518:     // RX Frequency
519:     if rbtUHF.Checked then
520:     begin
521:         if not ValidUHFFrequency(edtRXFrequency.Text) then
522:         begin
523:             MessageDlg ('Invalid RX Frequency', cstrInvalidUHFFrequencyMsg, mterror, [mbOk], 0);
524:             modalResult := mrNone;
525:             edtTXFrequency.Text := '';
526:             edtRXFrequency.SetFocus;
527:             Exit;
528:         end;
529:     end
530:     else
531:     begin
532:         if not ValidVHFFrequency(edtRXFrequency.Text) then
533:         begin
534:             MessageDlg ('Invalid RX Frequency', cstrInvalidVHFFrequencyMsg, mterror, [mbOk], 0);
535:             modalResult := mrNone;
536:             edtTXFrequency.Text := '';
537:             edtRXFrequency.SetFocus;

```

```

538:         Exit;
539:     end;
540: end;
541:
542: // DTSS Code
543: if chkDTSS.Checked then
544: begin
545:     if length (edtDTSSCode.Text) <> gcbytMaxDTSSCodeLength then
546:     begin
547:         ShowMessage(cstrInvalidDTSSCodeMsg);
548:         modalResult := mrNone;
549:         edtDTSSCode.SetFocus;
550:         Exit;
551:     end; // if length edtDTSSCode.Text <> gcbytMaxDTSSCodeLength
552: end; // if chkDTSS.Checked then
553:
554: // Channel Name
555: if length (edtChannelName.Text) < gcbytMinChannelNameLength then
556: begin
557:     ShowMessage(cstrInvalidChannelNameMsg);
558:     modalResult := mrNone;
559:     edtChannelName.SetFocus;
560:     Exit;
561: end; // if length (edtChannelName.Text) < gcbytMinChannelNameLength
562:
563: //=====
564: // Everything is valid so now we save the data in the appropriate array based on
565: // vdetDataEntryType.
566: //=====
567: case vdetDataEntryType of
568:
569:     detFAV : DataEntry_FAV_Save;
570:     detVFO : Begin
571:
572:         end;
573:
574:     detUHFMEM : DataEntry_UHFMEM_Save;
575:
576:     detVHFMEM : DataEntry_VHFMEM_Save;
577: end; // case vdetDataEntryType
578:
579: gvstrTMVDataChanged := True;
580:
581: end; // procedure TfrmDataEntry.bbtSaveClick
582:
583: //=====
584: // KEYPRESS ROUTINES
585: //=====
586: procedure TfrmDataEntry.edtRXFrequencyKeyPress(Sender: TObject; var Key: char);
587: begin
588:
589:     case Key of
590:
591:         #8 : Exit; // <BS>
592:         #46 : if Length(edtRXFrequency.Text) = 3 then // <.>
593:             Exit
594:         else
595:             begin
596:                 Key := #0;
597:                 Exit;

```

```
598:         end; // if Length(edtRXFrequency.Text) = 3
599:         #48..#57 : if (Length(edtRXFrequency.Text) < 3) or
600:             (Length(edtRXFrequency.Text) > 3) then // <.>
601:             Exit
602:         else
603:         begin
604:             Key := #0;
605:             Exit;
606:         end; // if Length(edtRXFrequency.Text) < 3
607:     else
608:         Key := #0;
609:     end; // case Key of
610:
611: end; // procedure TfrmDataEntry.edtRXFrequencyKeyPress
612:
613: //-----
614: procedure TfrmDataEntry.edtDTSSCodeKeyPress(Sender: TObject; var Key: char);
615: begin
616:
617:     case Key of
618:         #8 : Exit; // <BS>
619:         #48..#57 : Exit;
620:     else
621:         Key := #0;
622:     end; // case Key of
623:
624: end; // procedure TfrmDataEntry.edtDTSSKeyPress
625:
626: //-----
627: procedure TfrmDataEntry.edtChannelNameKeyPress(Sender: TObject; var Key: char);
628: begin
629:
630:     case Key of
631:         #8 : Exit; // <BS>
632:         #32 : Exit; // <Sp>
633:         #48..#57 : Exit; // <0>..<9>
634:         #65..#90 : Exit; // <A..Z>
635:         #97..#122 : Exit; // <a..z>
636:     else
637:         Key := #0;
638:     end; // case Key of
639:
640: end; // procedure TfrmDataEntry.edtChannelNameKeyPress
641:
642: //-----
643: procedure TfrmDataEntry.edtCommentsKeyPress(Sender: TObject; var Key: char);
644: begin
645:
646:     case Key of
647:         #8 : Exit; // <BS>
648:         #32 : Exit; // <Sp>
649:         #48..#57 : Exit; // <0>..<9>
650:         #65..#90 : Exit; // <A..Z>
651:         #97..#122 : Exit; // <a..z>
652:     else
653:         Key := #0;
654:     end; // case Key of
655:
656: end; // procedure TfrmDataEntry.edtCommentsKeyPress
657:
```

```
658: //=====
659: //          ON CHANGE ROUTINES
660: //=====
661: procedure TfrmDataEntry.rbtVHFChange(Sender: TObject);
662: begin
663:   SetShiftOffset;
664: end; // procedure TfrmDataEntry.rbtVHFChange(
665:
666: //-----
667: procedure TfrmDataEntry.rbtSimplexChange(Sender: TObject);
668: begin
669:   SetShiftOffset;
670: end; // procedure TfrmDataEntry.rbtSimplexChange
671:
672: //-----
673: procedure TfrmDataEntry.rbtPlusChange(Sender: TObject);
674: begin
675:   SetShiftOffset;
676: end; // procedure TfrmDataEntry.rbtPlusChange
677:
678: //-----
679: procedure TfrmDataEntry.rbtMinusChange(Sender: TObject);
680: begin
681:   SetShiftOffset;
682: end; // rocedure TfrmDataEntry.rbtMinusChange
683:
684: //-----
685: procedure TfrmDataEntry.rbtNoTonesChange(Sender: TObject);
686: begin
687:   SetToneFreq;
688: end; // procedure TfrmDataEntry.rbtNoTonesChange
689:
690: //-----
691: procedure TfrmDataEntry.chkDTSSChange(Sender: TObject);
692: begin
693:
694:   if chkDTSS.Checked then
695:   begin
696:     edtDTSSCode.Enabled := True;
697:     edtDTSSCode.Color := clWhite;
698:     edtDTSSCode.Text := gvstrFAVChannelDataArray[frmDataEntry.vbytChannelNumber,
699:       gcbytDTSSCodeField];
700:     edtDTSSCode.SetFocus;
701:   end
702:   else
703:   begin
704:     edtDTSSCode.Enabled := False;
705:     edtDTSSCode.Color := clYellow;
706:     edtDTSSCode.Text := '';
707:   end; // if chkDTSS.Checked
708:
709: end; // procedure TfrmDataEntry.chkDTSSChange
710:
711: //=====
712: //          ON EXIT ROUTINES
713: //=====
714: procedure TfrmDataEntry.edtRXFrequencyExit(Sender: TObject);
715: begin
716:
717:   if rbtVHF.Checked then
```

```
718: begin
719:     if not ValidVHFFrequency(edtRXFrequency.Text) then
720:         Exit;
721:     end
722: else
723:     begin
724:         if not ValidUHFFrequency(edtRXFrequency.Text) then
725:             Exit;
726:         end;// if rbtVHF.Checked
727:
728:         edtTXFrequency.Text := CalculateTXFrequency;
729:
730: end;// procedure TfrmDataEntry.edtRXFrequencyExit
731:
732: //=====
733: end.// unit DataEntry;
734:
```