```
1   { ***************************************************************************
2     *** Projet               : OneWayTickets                                 ***
3     *** Auteur               : Devaud Alan                                   ***
4     *** Description          : OneWay Tickets est un logiciel de billetteries***
5     ***                         séance                                       ***
6     *** Version              : 1.0                                           ***
7     *** Date de création     : 30.04.2015                                    ***
8     ***         Modification :                                               ***
9     ***         05.05.2015 - Ajout de l'accès au login administarteur        ***
10    ***                      Ajout l'accès au menu réservation en avance     ***
11    *************************************************************************** }
12  unit U_OneWayTicket;
13
14  interface
15
16  uses
17    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
18    Menus, StdCtrls, ExtCtrls, U_ImageBouton, IniFiles, U_FP, ShellAPI;
19
20  type
21    TFrmOneWayTickets = class(TForm)
22      MainMenu1: TMainMenu;
23      Fichier1: TMenuItem;
24      Quitter1: TMenuItem;
25      Administration1: TMenuItem;
26      N1: TMenuItem;
27      Connexion1: TMenuItem;
28      Aproposde1: TMenuItem;
29      BtnSeancePrecedente: TButton;
30      BtnSeanceSuivante: TButton;
31      BtnReserverAvance: TButton;
32      Timer1: TTimer;
33      lblHeureCourante: TLabel;
34      Label2: TLabel;
35      procedure Quitter1Click(Sender: TObject);
36      procedure Timer1Timer(Sender: TObject);
37      procedure FormCreate(Sender: TObject);
38      procedure BtnReserverAvanceClick(Sender: TObject);
39      procedure Connexion1Click(Sender: TObject);
40      procedure chargeToutesLesSeances();
41      procedure changeSecances(incremente: integer);
42      procedure BtnSeanceSuivanteClick(Sender: TObject);
43      procedure BtnSeancePrecedenteClick(Sender: TObject);
44      procedure destruction();
45      procedure FormActivate(Sender: TObject);
46    private
47      { Déclarations privées }
48    public
49      { Déclarations publiques }
50      procedure Reservation(film, horaire, salle, section : string; placesRestant:
        integer);
51      procedure Initialisation();
52    end;
53
54  Type
55    TListSeance = array of array [0..3] of String;
56
```

```
57  var
58    FrmOneWayTickets : TFrmOneWayTickets;
59    listImageBouton  : TList; // Liste qui contient tous les bouton
60    listSeances      : array of TSeance;
61    nbImageBouton    : integer = 11;
62    jourActuelle     : String;
63    prix             : array [0..2] of real;
64    temp             : integer;
65    secance          : integer = 0; // La sécances des séances a afficher
66
67  CONST
68    START               : integer = 10;
69    ECART_X             : integer = 10;
70    ECART_Y             : integer = 10;
71    TAILLE_WIDTH        : integer = 193;
72    TAILLE_HEIGHT       : integer = 57;
73    MDP_ADMIN           : String  = 'Neko1';
74    MAX_IMAGE_BOUTON    : integer = 11;
75    MAX_TICKETS_PAGE    : integer = 6;
76    CARAC_LIGNE_TICKET  : integer = 59;
77    TEMPS_ACTUALISATION : integer = 100;
78    MAX_BOUTON_LIGNE    : integer = 3;
79    NOMBRE_STATISTIQUE  : integer = 6;
80    NOMBRE_HORAIRE_MAX  : integer = 4;
81    CARAC_SEPAR_JOUR    : String = ',';
82    SECONDE_IMPRESSION  : integer = 20;
83
84
85
86  implementation
87
88  uses U_OneWayTickets_Reservation, U_OneWayTickets_ReservationAvance,
89    U_OneWayTickets_Login, U_OneWayTickets_MenuAdministrateur,
90    U_OneWayTickets_Impression;
91
92  {$R *.DFM}
93
94  { ************************************************************************
95    *** Transforme le texte en heure                                     ***
96    *** @params String horaire - Text a faire passer en heure            ***
97    *** @Result TTime - Heure                                            ***
98    ************************************************************************ }
99  function TextEnHeure(horaire : String): TTime;
100 var
101   h, m: word;
102 Begin
103   h:= StrToInt(horaire[1] + horaire[2]) ;
104   m:= StrToInt(horaire[4] + horaire[5]);
105   Result:= EncodeTime(h, m, 0, 0);
106 end;
107
108 { ************************************************************************
109   *** Charge les séances a afficher en séparant les heures             ***
110   *** @params array of TSeance - liste des séances                     ***
111   *** @Result TListSeance - Renvoi la liste des séances                ***
112   ************************************************************************ }
113 function chargeListDifferentHoraire(list: array of TSeance): TListSeance;
```

```
114   var
115     index, i, compteHoraire: integer;
116     ajout: boolean;
117     seances: TListSeance;
118   Begin
119     // Charge la liste des séances avec leurs heures différentes
120     index:= 0;
121
122     SetLength(seances, 1);
123     for i:= 0 to length(listSeances) - 1 do
124     Begin
125       if listSeances[i].film = '' then
126         break;
127
128       compteHoraire:= 0;
129
130       ajout:= true;
131       while ajout do
132       Begin
133
134         // Test si l'heure exist et si la séance n'est pas déjà passée
135         if (listSeances[i].heure1 <> '') and (TextEnHeure(listSeances[i].heure1) >
          TextEnHeure(TimeToStr(now()))) then
136         Begin
137           Seances[index][2]:= listSeances[i].section;
138           Seances[index][1]:= listSeances[i].heure1;
139           Seances[index][0]:= listSeances[i].film;
140           Seances[index][3]:= listSeances[i].salle;
141           inc(index);
142           inc(compteHoraire);
143           SetLength(seances, index + 1);
144         end
145         else
146           ajout:= false;
147
148         if (listSeances[i].heure2 <> '') and (TextEnHeure(listSeances[i].heure2) >
          TextEnHeure(TimeToStr(now()))) then
149         Begin
150           Seances[index][2]:= listSeances[i].section;
151           Seances[index][1]:= listSeances[i].heure2;
152           Seances[index][0]:= listSeances[i].film;
153           Seances[index][3]:= listSeances[i].salle;
154           inc(index);
155           inc(compteHoraire);
156           SetLength(seances, index + 1);
157         end
158         else
159           ajout:= false;
160
161         if (listSeances[i].heure3 <> '') and (TextEnHeure(listSeances[i].heure3) >
          TextEnHeure(TimeToStr(now()))) then
162         Begin
163           Seances[index][2]:= listSeances[i].section;
164           Seances[index][1]:= listSeances[i].heure3;
165           Seances[index][0]:= listSeances[i].film;
166           Seances[index][3]:= listSeances[i].salle;
167           inc(index);
```

```
168           inc(compteHoraire);
169           SetLength(seances, index + 1);
170         end
171         else
172           ajout:= false;
173
174         if (listSeances[i].heure4 <> '') and (TextEnHeure(listSeances[i].heure4) >
            TextEnHeure(TimeToStr(now()))) then
175         Begin
176           Seances[index][2]:= listSeances[i].section;
177           Seances[index][1]:= listSeances[i].heure4;
178           Seances[index][0]:= listSeances[i].film;
179           Seances[index][3]:= listSeances[i].salle;
180           inc(index);
181           inc(compteHoraire);
182           SetLength(seances, index + 1);
183         end
184         else
185           ajout:= false;
186
187         if compteHoraire mod NOMBRE_HORAIRE_MAX = 0 then
188           ajout:= false;
189       end;
190     end;
191
192     Result:= seances;
193   end;
194
195   { ***************************************************************************
196     *** Met à jour les statistique                                          ***
197     *************************************************************************** }
198   procedure statistique();
199   var
200     valeur: TValeur;
201     valeurs : TValeurs;
202     nbBilletTotal, i: integer;
203     OutPutList : TStringList;
204   begin
205     // Initialise les variables
206     nbBilletTotal:= 0;
207     SetLength(valeurs, NOMBRE_STATISTIQUE, 2);
208
209     //Récupère les informations du fichier de réservation
210     valeur:= lireFichier(FICHIER_RESERV);
211
212     // Calcul le nombre de billet vendu au total
213     for i:= 1 to length(valeur) - 1 do
214     Begin
215       if valeur[i] = '' then
216         Break;
217
218       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
219       nbBilletTotal:= nbBilletTotal + StrToInt(OutPutList[7]);
220
221     end;
222
223     valeurs[0][0]:= 'BilletTotal';
```

```
224     valeurs[0][1]:= IntToStr(nbBilletTotal);
225     nbBilletTotal:= 0;
226
227     // Calcul le nombre de billet vendu se mois
228     for i:= 1 to length(valeur) - 1 do
229     Begin
230       if valeur[i] = '' then
231         Break;
232
233        OutPutList:= Split(valeur[i], CARAC_SEPARATION);
234
235        if FormatDateTime(FORMAT_DATE, StrToDate(OutPutList[0])) = FormatDateTime(
           FORMAT_DATE, now) then
236          nbBilletTotal:= nbBilletTotal + StrToInt(OutPutList[7]);
237     end;
238
239     valeurs[1][0]:= 'BilletTotalMois';
240     valeurs[1][1]:= IntToStr(nbBilletTotal);
241     nbBilletTotal:= 0;
242
243     //Calcul le nombre de billet vendu par enfant
244     for i:= 1 to length(valeur) - 1 do
245     Begin
246       if valeur[i] = '' then
247         Break;
248
249       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
250       nbBilletTotal:= nbBilletTotal + StrToInt(OutPutList[3]);
251     end;
252
253     valeurs[2][0]:= 'BilletEnfants';
254     valeurs[2][1]:= IntToStr(nbBilletTotal);
255     nbBilletTotal:= 0;
256
257     // Calcul le nombre de billet vendu par adultes
258     for i:= 1 to length(valeur) - 1 do
259     Begin
260       if valeur[i] = '' then
261         Break;
262
263       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
264       nbBilletTotal:= nbBilletTotal + StrToInt(OutPutList[4]);
265     end;
266
267     valeurs[3][0]:= 'BilletAdults';
268     valeurs[3][1]:= IntToStr(nbBilletTotal);
269     nbBilletTotal:= 0;
270
271     // Calcul le nombre de billet vendu par EAA
272     for i:= 1 to length(valeur) - 1 do
273     Begin
274       if valeur[i] = '' then
275         Break;
276
277       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
278       nbBilletTotal:= nbBilletTotal + StrToInt(OutPutList[5]);
279     end;
```

```
280
281     valeurs[4][0]:= 'BilletEtudiants';
282     valeurs[4][1]:= IntToStr(nbBilletTotal);
283
284     ecritDansFichier(valeurs, FICHIER_STATS);
285   end;
286
287   { ************************************************************************
288     *** Détruit les objets et la liste                                   ***
289     ************************************************************************ }
290   procedure TFrmOneWayTickets.destruction();
291   var
292     i: integer;
293     bouton: TImageBouton;
294   Begin
295     // Parcoure la liste et détruit les objets
296     for i:= 0 to listImageBouton.Count - 1 do
297     Begin
298       bouton:= listImageBouton[i];
299       bouton.Destroy;
300       listImageBouton[i]:= bouton;
301     end;
302
303     listImageBouton.Destroy;
304   end;
305
306
307   { ************************************************************************
308     *** Change la secance de l'affichage des séances                     ***
309     ************************************************************************ }
310   procedure TFrmOneWayTickets.changeSecances(incremente: integer);
311   Begin
312     destruction();
313     secance:= secance + incremente;
314     Initialisation();
315   end;
316
317   { ************************************************************************
318     *** Génère les tickets                                               ***
319     *** @params String billet - Nom du type de billet (Enfant, adultes, etc.)***
320     *** @params Array of string resevatoin - Information sur la réservartoin ***
321     *** @params String salle - nom de la salle                           ***
322     *** @params array of real prix - Tableau des prix                    ***
323     *** @params String film - Nom du film                                ***
324     *** @Result String - Ticket sous forme ascii                         ***
325     ************************************************************************ }
326   function ticketAscii(billet: string; reservation: array of String; prix: real; film,
      salle: string): String;
327   var
328     ticket: string;
329   Begin
330     ticket:= #13#10 + #13#10 +
      '************************************************************' + #13#10
331               +  ajusterText('* Billet '+ billet, CARAC_LIGNE_TICKET) + '*' + #13#10 +
                   ajusterText('* Films  : '+ film, CARAC_LIGNE_TICKET) + '*' + #13#10
332                 + ajusterText('* Séance : ' + reservation[1] + ' - ' + reservation[0],
                     CARAC_LIGNE_TICKET) + '*' + #13#10
```

```
333                   + ajusterText('* Salle  : ' + salle, CARAC_LIGNE_TICKET) + '*' +
                      #13#10
334                     + ajusterText('* Prix   : ' + FloatToStr(prix) + ' ' + DEVISE,
                        CARAC_LIGNE_TICKET) + '*' + #13#10
335                       +
                          '************************************************************';
336
337     Result:= ticket;
338   end;
339
340   { ***************************************************************************
341     *** Génère les tickets                                                  ***
342     *** @params Array of string resevatoin - Information sur la réservartoin ***
343     *** @params String film - Nom du film                                   ***
344     *** @params String salle - nom de la salle                              ***
345     *** @params array of real prix - Tableau des prix                       ***
346     *************************************************************************** }
347   procedure generationTickets(reservation: array of String; film, salle: string; prix:
      array of real);
348   var
349     nbBilletEnfant, nbBilletAdulte, nbBilletEAA,nbBilletExistant: integer;
350     tickets: TValeurs;
351   Begin
352     // Initialisation des paramètres
353     SetLength(tickets, StrToInt(reservation[3]) + StrToInt(reservation[4]) + StrToInt(
      reservation[5]) + 1, 1);
354     nbBilletEnfant:= 0 ;
355     nbBilletAdulte:= 0;
356     nbBilletEAA:= 0;
357     nbBilletExistant:= 0;
358
359     // Génération des billets enfants
360     while nbBilletEnfant < StrToInt(reservation[3]) do
361     Begin
362       tickets[nbBilletEnfant + nbBilletExistant][0]:= ticketAscii('enfant', reservation
        , prix[1], film, salle);
363       inc(nbBilletEnfant);
364     end;
365
366     nbBilletExistant:= nbBilletEnfant + nbBilletExistant;
367
368     // Génération des billets adultes
369     while nbBilletAdulte < StrToInt(reservation[4]) do
370     Begin
371       tickets[nbBilletAdulte + nbBilletExistant][0]:= ticketAscii('adulte', reservation
        , prix[0], film, salle);
372       inc(nbBilletAdulte);
373     end;
374
375     nbBilletExistant:= nbBilletAdulte + nbBilletExistant;
376
377     // Génération des billets etudiant, a.v.s et a.v.i
378     while nbBilletEAA < StrToInt(reservation[5]) do
379     Begin
380       tickets[nbBilletEAA + nbBilletExistant][0]:= ticketAscii('Etudiant / A.V.S /
        A.V.I', reservation, prix[2], film, salle);
381       inc(nbBilletEAA);
```

```
382     end;
383
384     // Ecrit dans le fichier
385     if ecritDansFichier(tickets, FICHIER_TICKETS) then
386     Begin
387       // Imprime le billet
388       ShellExecute(FrmOneWayTickets.Handle, 'Open',Pchar('NotePad'), PChar('/p ' + '"'
        + FICHIER_TICKETS + '"'), nil, SW_HIDE);
389
390       // Affiche "Impression en cours" et initilalise les valeurs du timer
391       FrmImpressionEnCours.visible:= true;
392       FrmImpressionEnCours.maxSecond:= SECONDE_IMPRESSION;
393       FrmImpressionEnCours.second:= 0;
394       FrmImpressionEnCours.Timer1.Enabled:= true;
395     end;
396   end;
397
398   { ***************************************************************************
399     *** Ouvre la fenêtre de réservation                                     ***
400     *************************************************************************** }
401   procedure TFrmOneWayTickets.Reservation(film, horaire, salle, section : string;
      placesRestant: integer);
402   var
403     OutPutList: TStringList;
404     NbBilletsEnfants, NbBilletsEAA, NbBilletsAdultes, PrixTotal, Date: String;
405     places: string;
406     valeur: TValeur;
407     i: integer;
408     reservations: array [0..7] of String;
409   Begin
410
411     // Charge les informations du film
412     valeur:= lireFichier(FICHIER_FILMS);
413     for i:= 0 to length(valeur) - 1 do
414     Begin
415       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
416       if OutPutList[0] = film then
417       Begin
418         FrmReservation.lblNomFilm.Caption:= film;
419         FrmReservation.lblDuree.Caption:= OutPutList[1];
420         FrmReservation.mmoSynopsis.Clear;
421         FrmReservation.mmoSynopsis.Text:= OutPutList[2];
422         Break;
423       end;
424     end;
425
426     // Charge les informations de la séacne
427     valeur:= lireFichier(FICHIER_SALLES);
428     for i:= 0 to length(valeur) - 1 do
429     Begin
430       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
431       if OutPutList[0] = salle then
432       Begin
433         FrmReservation.lblSalle.Caption:= salle;
434         FrmReservation.lblHoraire.Caption:= horaire;
435         FrmReservation.lblPlacesRestantes.Caption:= '0';
436         Break;
```

```
437       end;
438     end;
439
440     // Charge les prix
441     FrmReservation.lblPrixEnfants.Caption:= 'x ' + FloatToStr(prix[1]) + ' ' + DEVISE;
442     FrmReservation.lblPrixAdultes.Caption:= 'x ' + FloatToStr(prix[0]) + ' ' + DEVISE;
443     FrmReservation.lblPrixEAA.Caption:= 'x ' + FloatToStr(prix[2]) + ' ' + DEVISE;
444     FrmReservation.PrixEnfant:= prix[1];
445     FrmReservation.PrixEAA:= prix[2];
446     FrmReservation.PrixAdulte:= prix[0];
447     FrmReservation.prixTotal:= prix[0];
448     FrmReservation.calculTotal();
449
450     // Initialisation des champs
451     FrmReservation.EdtNbBilletsEnfants.Text:= '0';
452     FrmReservation.EdtNbBilletsAdultes.Text:= '1';
453     FrmReservation.EdtNbBilletsEAA.Text:= '0';
454     FrmReservation.lblPlacesRestantes.Caption:= IntToStr(placesRestant);
455
456     if FrmReservation.ShowModal = mrOk then
457     Begin
458       // Récupération des éléments
459       NbBilletsEnfants:= FrmReservation.EdtNbBilletsEnfants.Text;
460       NbBilletsAdultes:= FrmReservation.EdtNbBilletsAdultes.Text;
461       NbBilletsEAA:= FrmReservation.EdtNbBilletsEAA.Text;
462       PrixTotal:= FloatToStr(FrmReservation.prixTotal);
463       Date:= DateToStr(now);
464       places:= IntToStr(StrToInt(NbBilletsEnfants) + StrToInt(NbBilletsAdultes)
465         + StrToInt(NbBilletsEAA));
466
467       // Initialise le tableau de valeur
468       reservations[0]:= Date;
469       reservations[1]:= horaire;
470       reservations[2]:= section;
471       reservations[3]:= NbBilletsEnfants;
472       reservations[4]:= NbBilletsAdultes;
473       reservations[5]:= NbBilletsEAA;
474       reservations[6]:= PrixTotal;
475       reservations[7]:= places;
476
477       if ajoutUneLigne(reservations, FICHIER_RESERV) then
478         MessageDlg('Reservation effectuée avec succés !', mtInformation, [mbOk,
          mbCancel], 0)
479       else
480         MessageDlg('Une erreur est survenue lors de la réservation !', mtError, [mbOk,
          mbCancel], 0);
481
482       generationTickets(reservations, FrmReservation.lblNomFilm.Caption, FrmReservation
          .lblSalle.Caption, prix);
483
484       // Met a jour les statistiques
485       statistique();
486
487     end;
488   end;
489
490
```

```
491   { ***************************************************************************
492     *** Charge toutes les séances depuis le fichier ini                     ***
493     *************************************************************************** }
494   procedure TFrmOneWayTickets.chargeToutesLesSeances();
495   var
496     i, j, index: integer;
497     fichierIni: TIniFile;
498     Sections, dataSections : TStringList;
499     listjour : TStringList;
500     jourDiff: String;
501   Begin
502     fichierIni:= TIniFile.Create(FICHIER_SEANCES);
503     Sections:= TStringList.Create;
504     dataSections:= TStringList.Create;
505     index:= 0;
506
507     fichierIni.ReadSections(Sections);
508
509     // Parcoure toutes les sections
510     for i:= 0 to Sections.Count - 1 do
511     Begin
512
513       fichierIni.ReadSection(Sections[i], dataSections);
514       jourDiff:= fichierIni.ReadString(Sections[i], dataSections[2], 'N/A');
515
516       jourDiff:= assembleJour(jourDiff, CARAC_SEPAR_JOUR);
517       listJour:= Split(jourDiff, CARAC_SEPAR_JOUR);
518
519       for j:= 0 to listjour.count - 1 do
520       Begin
521         // Test si le jour correspond et ajoute la séance dans la liste des séances
522         if listJour[j] = jourActuelle then
523         Begin
524           listSeances[index].section:= Sections[i];
525           listSeances[index].film:= fichierIni.ReadString(Sections[i], dataSections[0],
              'N/A');
526           listSeances[index].salle:= fichierIni.ReadString(Sections[i], dataSections[1
              ], 'N/A');
527           listSeances[index].jourDiff:= fichierIni.ReadString(Sections[i], dataSections
              [2], 'N/A');
528           listSeances[index].heure1:= fichierIni.ReadString(Sections[i], dataSections[3
              ], 'N/A');
529           listSeances[index].heure2:= fichierIni.ReadString(Sections[i], dataSections[4
              ], 'N/A');
530           listSeances[index].heure3:= fichierIni.ReadString(Sections[i], dataSections[5
              ], 'N/A');
531           listSeances[index].heure4:= fichierIni.ReadString(Sections[i], dataSections[6
              ], 'N/A');
532           listSeances[index].diffuser:= fichierIni.ReadString(Sections[i], dataSections
              [7], 'N/A');
533           inc(index);
534           break;
535         end;
536       end;
537     end;
538
539     setLength(listSeances, index);
```

```
540     fichierIni.free;
541   end;
542
543   { ***************************************************************************
544     *** Initialisation de l'affichage                                       ***
545     *************************************************************************** }
546   procedure TFrmOneWayTickets.Initialisation();
547   var
548     i, x, y, index, j: integer;
549     Seances: TListSeance;
550     OutPutList: TStringList;
551     nbPlacesTotal, nbPlacesVendu, nbPlacesRestant, nbLigne: Integer;
552     bouton: TImageBouton;
553     valeur: TValeur;
554   Begin
555
556     jourActuelle:= FormatDateTime('dddd', now());
557     temp:= 0; // Recommence le compte
558     nbPlacesTotal:= 0;
559     i:= 0;
560     nbLigne:= nbLignesFichier(FICHIER_SEANCES) div NOMBRE_SECTIONS; // Charge le
        nombre de séances (nombre de ligne diviser par le nombre de données dans les
        sections)
561     // Initialisation des seances
562     SetLength(listSeances, nbLigne);
563     chargeToutesLesSeances();
564     // Charge la liste des séances avec leurs heures différentes
565     Seances:= chargeListDifferentHoraire(listSeances);
566
567     //Initialisation des boutons
568     x:= START; // Position X de départ du bouton
569     y:= START; // Position Y de départ du bouton
570     index:= secance;
571
572     // Crée la liste des boutons
573     listImageBouton:= TList.Create();
574     nbImageBouton:= length(Seances);
575
576     for i:= 0 to MAX_IMAGE_BOUTON do
577     Begin
578       // Test s'il y a encore des séances à afficher
579       if index = nbImageBouton - 1 then
580       Begin
581         BtnSeanceSuivante.Enabled:= false;
582         Break;
583       end
584       else
585         BtnSeanceSuivante.Enabled:= true;
586
587       bouton:= TImageBouton.Create(FrmOneWayTickets, seances[index][0], seances[index][
          1], seances[index][2], seances[index][3], x, y);
588
589       // Ouvre le fichier des salles pour récupèrer le nombre de place
590       valeur:= lireFichier(FICHIER_SALLES);
591       for j:= 0 to length(valeur) - 1 do
592       Begin
593         if valeur[j] = '' then
```

```
594           Break;
595         OutPutList:= Split(valeur[j], CARAC_SEPARATION);
596         if OutPutList[0] = seances[index][3] then
597         Begin
598           nbPlacesTotal:= StrToInt(OutPutList[1]);
599           Break;
600         end;
601       end;
602
603       // Ouvre le fichier des reservation pour récupèrer le nombre de place vendu
604       nbPlacesVendu:= 0;
605       valeur:= lireFichier(FICHIER_RESERV);
606       for j:= 0 to length(valeur) - 1 do
607       Begin
608         if valeur[j] = '' then
609           Break;
610
611         OutPutList:= Split(valeur[j], CARAC_SEPARATION);
612         if (OutPutList[0] = DateToStr(now)) and (OutPutList[1] = seances[index][1])
613           and (OutPutList[2] = seances[index][2]) then
614         Begin
615           nbPlacesVendu:= StrToInt(OutPutList[7]) + nbPlacesVendu;
616         end;
617       end;
618
619       // Calcule le nombre de place restante
620       nbPlacesRestant:= nbPlacesTotal - nbPlacesVendu;
621       bouton.pPlaces:= nbPlacesRestant;
622
623       if nbPlacesRestant <= 0 then
624       begin
625         bouton.Complet();
626       end;
627
628       listImageBouton.add(bouton);
629
630
631       x := x + TAILLE_WIDTH + ECART_X; // Déplace le prochain bouton sur la droite
632
633       // Regare s'il y a déjà trois bouton sur une ligne
634       if listImageBouton.Count mod MAX_BOUTON_LIGNE = 0 then
635       Begin
636         y:= y + TAILLE_HEIGHT + ECART_X; // Déplace les prochain bouton sur la seconde
          ligne
637         x:= START; // Redémare la ligne au début
638       end;
639
640       inc(index);
641     end;
642
643     // Test le nombre de bouton qu'il y a pour activé le bouton des séances
644     // suivantes
645     if index < MAX_IMAGE_BOUTON - 1 then
646       BtnSeanceSuivante.Enabled:= false;
647
648     // Récupère les prix
649     valeur:= lireFichier(FICHIER_PRIX);
```

```
650     OutPutList:= TStringList.Create;
651     for i:= 0 to length(valeur) - 1 do
652     Begin
653       if valeur[i] = '' then
654           Break;
655
656       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
657       prix[i]:= StrToInt(OutPutList[1]);
658       prix[i]:= StrToInt(OutPutList[1]);
659       prix[i]:= StrToInt(OutPutList[1]);
660     end;
661     OutPutList.Free;
662
663     statistique();
664   end;
665
666   { ***************************************************************************
667     *** Quitter l'application                                                ***
668     *************************************************************************** }
669   procedure TFrmOneWayTickets.Quitter1Click(Sender: TObject);
670   begin
671     close;
672   end;
673
674   { ***************************************************************************
675     *** Affiche l'heure courante                                             ***
676     *************************************************************************** }
677   procedure TFrmOneWayTickets.Timer1Timer(Sender: TObject);
678   begin
679     lblHeureCourante.Caption:= TimeToStr(now());
680
681     if temp = TEMPS_ACTUALISATION then
682     Begin
683       destruction();
684       Initialisation();
685     end
686     else
687       inc(temp);
688   end;
689
690   { ***************************************************************************
691     *** Evènement au démarrage de l'application                              ***
692     *************************************************************************** }
693   procedure TFrmOneWayTickets.FormCreate(Sender: TObject);
694   begin
695     Initialisation();
696   end;
697
698   { ***************************************************************************
699     *** Réservation en avance                                                ***
700     *************************************************************************** }
701   procedure TFrmOneWayTickets.BtnReserverAvanceClick(Sender: TObject);
702   begin
703     FrmReservationAvance.CbxFilm.ItemIndex:= 0;
704     FrmReservationAvance.showModal;
705   end;
706
```

```
707   { ***************************************************************************
708     *** Permet la connexion en tant qu'administrateur et initialise les      ***
709     *** données de statistiques                                              ***
710     *************************************************************************** }
711   procedure TFrmOneWayTickets.Connexion1Click(Sender: TObject);
712   var
713     mdp : string;
714     OutPutList : TStringList;
715     valeur : TValeur;
716     i: integer;
717   begin
718     // Vide le champ text du mot de passe
719     FrmLogin.edtMdpAdmin.Text:= '';
720     valeur:= nil;
721
722     // Traitement de la fenêtre à la validation de la connexion
723     if FrmLogin.showModal = mrOk then
724     Begin
725       mdp:= FrmLogin.edtMdpAdmin.Text;
726
727       // Test si le mot de passe est valide
728       if mdp = MDP_ADMIN then
729       Begin
730         valeur:= lireFichier(FICHIER_STATS);
731         OutPutList:= TStringList.create;
732         for i:= 0 to length(valeur) - 1 do
733         Begin
734           if valeur[i] = '' then
735             break;
736
737           OutPutList:= Split(valeur[i], ';');
738           // Initialisation des labels du menu administrateur
739           // Test quel label il faut mettre à jour
740           if OutPutList[0] = 'BilletTotal' then
741             FrmMenuAdministrateur.lblBilletTotalVendu.Caption:= OutPutList[1];
742
743           if OutPutList[0] = 'BilletEnfants' then
744             FrmMenuAdministrateur.lblBilletEnfantsVendu.Caption:= OutPutList[1];
745
746           if OutPutList[0] = 'BilletAdults' then
747             FrmMenuAdministrateur.lblBilletAdultesVendu.Caption:= OutPutList[1];
748
749           if OutPutList[0] = 'BilletEtudiants' then
750             FrmMenuAdministrateur.lblBilletEtudiantsVendu.Caption:= OutPutList[1];
751
752           if OutPutList[0] = 'BilletTotalMois' then
753             FrmMenuAdministrateur.lblBilletVenduMoisCourrant.Caption:= OutPutList[1];
754
755           if OutPutList[0] = 'FilmPlusVu' then
756             FrmMenuAdministrateur.lblFilmPlusVu.Caption:= OutPutList[1];
757         end;
758         OutPutList.Free;
759
760         FrmOneWayTickets.Visible:= false;
761         FrmMenuAdministrateur.Visible:= true; // Affiche la fenêtre administrateur
762
763       end
```

```
764       else
765       Begin
766         // Message d'erreur
767         MessageDlg('Connexion refusée !' + #13#10 + 'Mot de passe administrateur
            incorrect !', mtError, [mbOk, mbCancel], 0);
768       end;
769     end;
770   end;
771
772   { ***************************************************************************
773     *** Procedure pour voir les séances futur                               ***
774     *************************************************************************** }
775   procedure TFrmOneWayTickets.BtnSeanceSuivanteClick(Sender: TObject);
776   begin
777     changeSecances(MAX_IMAGE_BOUTON + 1);
778
779     if secance > 0 then
780       BtnSeancePrecedente.Enabled:= true
781   end;
782
783   { ***************************************************************************
784     *** Procedure pour afficher les séances précédente                      ***
785     *************************************************************************** }
786   procedure TFrmOneWayTickets.BtnSeancePrecedenteClick(Sender: TObject);
787   begin
788     changeSecances((MAX_IMAGE_BOUTON + 1) * (-1));
789
790     if secance = 0 then
791       BtnSeancePrecedente.Enabled:= false;
792   end;
793
794   { ***************************************************************************
795     *** Procedure lors de l'activation de la fenêtre                        ***
796     *************************************************************************** }
797   procedure TFrmOneWayTickets.FormActivate(Sender: TObject);
798   begin
799     destruction();
800     Initialisation();
801   end;
802
803   end.
804
```

```
1  { ***************************************************************************
2    *** Projet            : OneWayTickets                                   ***
3    *** Unité             : U_FP                                            ***
4    *** Auteur            : Devaud Alan                                     ***
5    *** Description       : Regroupe toutes les fonctions utilisable        ***
6    *** Version           : 1.0                                             ***
7    *** Date de création  : 05.05.2015                                      ***
8    *************************************************************************** }
9  unit U_FP;
10
11 interface
12   uses
13     classes, SysUtils, StdCtrls;
14
15 type
16   TValeurs = array of array of string;
17   TValeur  =  array of String;
18
19   function Split(chaine: String; delimiteur: string): TStringList;
20   function assembleJour(jour, caracSeparation : String): String;
21   function AjoutEspace(tailleEspace: integer): string;
22   function AjusterText(text: string; taille: integer): string;
23   function ecritDansFichier(elements: TValeurs; fichier: string):Boolean;
24   function ajoutUneLigne(elements: array of String; fichier: string): boolean;
25   function lireFichier(fichier: String): TValeur;
26   function nbLignesFichier(fichier: string): integer;
27   function valeurExists(fichier, valeur: string): boolean;
28
29 // Information sur les séances
30 Type
31   TSeance = record
32     section  : String;
33     film     : String;
34     salle    : String;
35     jourDiff : String;
36     heure1   : String;
37     heure2   : String;
38     heure3   : String;
39     heure4   : String;
40     diffuser : String;
41   end;
42
43 // Information sur un film
44 Type
45   TFilm = record
46     NomFilm  : String;
47     Duree    : String;
48     Prix     : String;
49     Synopsis : String;
50   end;
51
52   // Information sur une salle
53 Type
54   TSalle = Record
55     NomSalle : String;
56     Places   : String;
57   end;
```

```
58
59  const
60    FICHIER_STATS    : String  = './Res/statistiques.csv';
61    FICHIER_SEANCES  : String  = './Res/seances.ini';
62    FICHIER_FILMS    : String  = './Res/films.csv';
63    FICHIER_SALLES   : String  = './Res/salles.csv';
64    FICHIER_PRIX     : String  = './Res/prixBillets.csv';
65    FICHIER_RESERV   : String  = './Res/reservations.csv';
66    FICHIER_TICKETS  : String  = './Res/tickets.txt';
67    DEVISE           : String  = 'CHF';
68    NOMBRE_SECTIONS  : Integer = 9;
69    CARAC_SEPARATION : String  = ';';
70    FORMAT_DATE      : String  = 'mmmm';
71    CARAC_TEXT_SUITE : String  = '...';
72
73
74
75  implementation
76
77  { ***************************************************************************
78    *** Split une chaine de caractère                                       ***
79    *** Code trouvé sur le web, utilisateur qui donne le code : Aos         ***
80    *** http://www.developpez.net/forums/d639683/environnements-developpement***
81    *** /delphi/debutant/fonction-split-delphi/                             ***
82    *** @params String chaine - Chaine a splitée                            ***
83    *** @params String delimiteur - Délimiteur pour le split               ***
84    *** @Result TStringList - List des éléments splité                      ***
85    *************************************************************************** }
86  function Split(chaine: String; delimiteur: string): TStringList;
87  var
88    L : TStringList;
89  Begin
90    L:= TStringList.Create;
91    L.Text:= StringReplace(chaine, delimiteur, #13#10, [rfReplaceAll]);
92    Split:= L;
93  end;
94
95  { ***************************************************************************
96    *** Traduit les jours de diffusion (1 -> Lundi, etc.)                   ***
97    *** @params String jour - Suite de nombre qui compose les jours         ***
98    *** @params String caracSeparation - Caracère qui sépare les jours      ***
99    *** @Result string - Chaine comportant les jours                        ***
100   *************************************************************************** }
101 function assembleJour(jour, caracSeparation : String): String;
102 var
103   j, tmpJour: integer;
104   jourDiff: string;
105 Begin
106   jourDiff:= '';
107
108   // Boucle qui parcoure tous les jours
109   for j:= 1 to length(jour) do
110   Begin
111     tmpJour:= StrToInt(jour[j]); // Récupère la valeur au point j et le traduit en
      integer
112     case tmpJour of
113       1 : jourDiff:= jourDiff + 'lundi';
```

```
114         2 : jourDiff:= jourDiff + 'mardi';
115         3 : jourDiff:= jourDiff + 'mercredi';
116         4 : jourDiff:= jourDiff + 'jeudi';
117         5 : jourDiff:= jourDiff + 'vendredi';
118         6 : jourDiff:= jourDiff + 'samedi';
119         7 : jourDiff:= jourDiff + 'dimanche';
120       end;
121
122       if j < length(jour) then
123         jourDiff:= jourDiff + caracSeparation;
124     end;
125
126     Result:= jourDiff;
127   end;
128
129   { ************************************************************************
130     *** Ajout un nombre d'espace avant ou après le text                   ***
131     *** @params Integer tailleEspace - Nombre d'espace souhaité           ***
132     *** @Result string - Retourne le text modifié                         ***
133     ************************************************************************ }
134   function AjoutEspace(tailleEspace: integer): string;
135   var
136     i : integer;
137     text: string;
138   Begin
139     text:= '';
140     // Ajoute ce qu'il faut
141     for i:= 0 to tailleEspace do
142     Begin
143       text:= text + ' ';
144     end;
145
146     Result:= text; // Retourne le text modifier
147   end;
148
149   { ************************************************************************
150     *** Ajuste la taille du text                                          ***
151     *** @params String text - Text qui va être ajusté                     ***
152     *** @params Integer taille - Taille final souhaitée                   ***
153     *** @Result string - Retourne le text modifié                         ***
154     ************************************************************************ }
155   function AjusterText(text: string; taille: integer): string;
156   var
157     difference: integer;
158   Begin
159
160     if length(text) > taille then
161     Begin
162       text:= copy(text, 1, taille - length(CARAC_TEXT_SUITE));
163       text:= text + CARAC_TEXT_SUITE;
164     end;
165
166     if length(text) < taille then
167     Begin
168       difference:= taille - length(text);
169       text:= text + AjoutEspace(difference);
170     end;
```

```
171
172     Result:= text;
173   end;
174
175   { ************************************************************************
176     *** Ecrit dans le fichier text                                       ***
177     *** @params TValeurs elements - Liste des valeurs qui seront écrit   ***
178     *** @params string fichier - Chemin du fichier                       ***
179     *** @Result Boolean - Retourne vrai si l'écriture c'est bien passée  ***
180     ************************************************************************ }
181   function ecritDansFichier(elements: TValeurs; fichier: string):Boolean;
182   var
183     f: TextFile;
184     element: string;
185     index, i: integer;
186     success: boolean;
187   Begin
188     success:= false;
189     element:= '';
190
191     if FileExists(fichier) then
192     Begin
193       AssignFile(f, fichier);
194       rewrite(f);
195       index:= 0;
196
197       // Parcoure le tableau et écrit les éléments dans le fichier
198       while (elements[index][0] <> '') and (index < length(elements) - 1) do
199       Begin
200         element:= '';
201         for i:= 0 to length(elements[index]) - 1 do
202         Begin
203           element:= element + elements[index][i];
204
205           if i < length(elements[index]) - 1 then
206             element:= element + CARAC_SEPARATION;
207         end;
208         writeln(f, element);
209         inc(index);
210       end;
211
212       CloseFile(f);
213
214       success:= true;
215     end;
216
217     Result:= success;
218   end;
219
220   { ************************************************************************
221     *** ajoute une ligne dans le fichier text                            ***
222     *** @params array of string elements - Liste des éléments qui sont écrit ***
223     *** @params string fichier - Chemin du fichier                       ***
224     *** @Result Boolean - Retourne vrai si l'écriture c'est bien passée  ***
225     ************************************************************************ }
226   function ajoutUneLigne(elements: array of string; fichier: string): boolean;
227   var
```

```
228     f: TextFile;
229     element: String;
230     i: integer;
231     success: boolean;
232   begin
233     success:= false;
234     // Initialise la ligne qui sera enregistrée dans le fichier
235     for i:= 0 to length(elements) - 1 do
236     Begin
237       element:= element + elements[i];
238
239       if i < length(elements) - 1 then
240         element:= element + ';';
241     end;
242
243     // Test si le fichier existe
244     if FileExists(fichier) then
245     Begin
246       AssignFile(f, fichier);
247       Append(f);
248       Writeln(f, element);
249       CloseFile(f);
250       success:= true;
251     end;
252
253     Result:= success;
254   end;
255
256   { ************************************************************************
257     *** Lit un fichier text                                              ***
258     *** @params string fichier - Chemin du fichier                       ***
259     *** @Result Boolean - Retourne les valeurs chargées depuis le fichier    ***
260     ************************************************************************ }
261   function lireFichier(fichier: String): TValeur;
262   var
263     f: TextFile;
264     i, nbLigne: integer;
265     valeur: TValeur;
266   Begin
267     nbLigne:= nbLignesFichier(fichier);
268     SetLength(valeur, nbLigne);
269     // Test si le fichier existe
270     if FileExists(fichier) then
271     Begin
272       // Assigne le fichier
273       AssignFile(f, fichier);
274       Reset(f);
275       i:= 0;
276
277       repeat
278         Readln(f, valeur[i]);
279         inc(i);
280       until eof(f);
281
282       CloseFile(f);
283     end;
284
```

```
285     Result:= valeur;
286   end;
287
288   { ************************************************************************
289     *** Compte le nombre de ligne dans un fichier                          ***
290     *** @params String fichier - Chemin du fichier                         ***
291     *** @Result integer - renvoi le nombre de ligne d'un fichier           ***
292     ************************************************************************ }
293   function nbLignesFichier(fichier: string): integer;
294   var
295     SL: TStringList;
296   Begin
297     SL:= TStringList.Create;
298     SL.LoadFromFile(fichier);
299     Result:= SL.Count;
300     FreeAndNil(SL);
301   end;
302
303
304   { ************************************************************************
305     *** Vérifie si l'élément exist dans le fichier                         ***
306     *** @params String fichier - Chemin du fichier                         ***
307     *** @params String valeur - Valeur à chercher dans le fichier          ***
308     *** @Result integer - renvoi vrai si la valeur est trouvée            ***
309     ************************************************************************ }
310   function valeurExists(fichier, valeur: string): boolean;
311   var
312     lignes: TValeur;
313     i, j: integer;
314     exist: boolean;
315     OutPutList: TSTringList;
316   Begin
317     //Initialise les variables et récupère les données du fichier
318     lignes:= lireFichier(fichier);
319     exist:= false;
320
321     // Tourne dans chaque lignes
322     for i:= 0 to length(lignes) - 1 do
323     Begin
324       // Crée une TStringList pour récupérer les éléments d'une ligne
325       OutPutList:= TStringList.Create;
326       OutPutList:= Split(lignes[i], CARAC_SEPARATION);
327
328
329       // Tourne dans la lsite
330       for j:= 0 to OutPutList.Count - 1 do
331       Begin
332         // Vérifie si la valeur de la liste[i] vaut la valeur à chercher
333         if UpperCase(OutPutList[j]) = UpperCase(valeur) then
334           exist:= true;
335
336       end;
337
338       // Test si existe est vrai
339       if exist then
340         Break;
341     end;
```

```
342
343     Result:= exist;
344
345   end;
346
347   end.
348
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                  ***
3    *** Auteur               : Devaud Alan                                    ***
4    *** Description          : ImageBouton permet de créer un bouton coloré   ***
5    ***                       qui contient quelques informations sur la       ***
6    ***                       séance                                          ***
7    *** Version              : 1.0                                            ***
8    *** Date de création     : 30.04.2015                                     ***
9    ***         Modification :                                                ***
10   ***         07.05.2015 - AD - Modification du constructeur (section)      ***
11   ***         08.05.2015 - AD - Ajout de l'etat complet                     ***
12   *************************************************************************** }
13 unit U_ImageBouton;
14
15 interface
16
17 uses
18   ExtCtrls, StdCtrls, Classes, Forms, graphics;
19
20 Type
21   TImageBouton = Class(TImage)
22   private
23     lblNomFilm   : TLabel;
24     lblHoraire   : TLabel;
25     section      : String;
26     salle        : string;
27     complet_     : boolean;
28     places       : integer;
29   public
30     constructor Create(AOwner: TComponent; nomFilm, horaire, _section, _salle :
       String; x, y : integer); overload;
31     procedure changeCouleur(couleur: TColor);
32     procedure setText(texte: string; objet: string);
33     procedure Click(Sender: TObject);
34     procedure Complet();
35     property pSection : string read section write section;
36     property pSalle : String read salle write salle;
37     property pPlaces : Integer read places write places;
38     destructor Destroy; overload;
39   end;
40
41 CONST
42   WIDTH_IMAGE   : integer = 193;
43   HEIGHT_IMAGE  : integer = 57;
44   MIN_CARACTERE : integer = 1;
45   MAX_CARACTERE : integer = 30;
46
47 implementation
48
49 uses
50   U_OneWayTickets_Reservation, U_OneWayTicket;
51
52 { ***************************************************************************
53   *** Constructeur de ImageBouton                                         ***
54   *** Crée et initialise le boutonImage                                   ***
55   *** @params TComponent AOwner                                           ***
56   *** @params String nomFilm - Nom du film à afficher                     ***
```

```
57      *** @params String horaire - horaire du film à afficher                  ***
58      ********************************************************************************* }
59    constructor TImageBouton.Create(AOwner : TComponent; nomFilm, horaire, _section,
      _salle: String; x, y : integer);
60    Begin
61      inherited Create(AOwner);
62
63      // Initialise les propriétés du bouton
64      with self do
65      Begin
66        Parent:= (AOwner as TForm);
67        Width:= WIDTH_IMAGE;
68        Height:= HEIGHT_IMAGE;
69        Top:= y;
70        Left:= x;
71        pSection:= _section;
72        pSalle:= _salle;
73        complet_:= false;
74      end;
75
76
77      self.changeCouleur(clGreen);
78
79      // Initialise le label du nom du film
80      lblNomFilm:= TLabel.Create(self.Parent);
81      with lblNomFilm do
82      Begin
83        Parent:= (self.Parent as TForm);
84        Transparent:= true;
85        AutoSize:= false;
86
87        // Taille et placement
88        Width:= self.Width;
89        left:= self.Left;
90        Top:= self.Top + 4;
91        Alignment:= taCenter;
92
93        // Modification de la police
94        Font.Size:= 10;
95        Font.Style:= [fsBold];
96        Font.Name:= 'Arial';
97      end;
98
99      // Initialise le label de l'horaire
100     lblHoraire:= TLabel.Create(self.Parent);
101     with lblHoraire do
102     Begin
103       Parent:= (self.Parent as TForm);
104       Transparent:= true;
105       AutoSize:= false;
106
107       // Taille et placement
108       Width:= self.Width;
109       left:= self.Left;
110       Top:= lblNomFilm.Top + 24;
111       Alignment:= taCenter;
112
```

```
113       // Modification de la police
114       Font.Size:= 8;
115       Font.Name:= 'Arial';
116     end;
117
118     // Initialise le texte des champs
119     self.setText(horaire, 'horaire');
120     self.setText(nomFilm, 'nom');
121
122     // Ajout d'évènement
123     OnClick:= Click;
124     lblNomFilm.OnClick:= Click;
125     lblHoraire.OnClick:= Click;
126   end;
127
128   { ************************************************************************
129     *** Destructeur de l'objet                                           ***
130     ************************************************************************ }
131   destructor TImageBouton.Destroy();
132   Begin
133     lblNomFilm.Destroy;
134     lblHoraire.Destroy;
135     inherited Destroy;
136   end;
137
138   { ************************************************************************
139     *** Procedure si la séance est complet                               ***
140     ************************************************************************ }
141   procedure TImageBouton.Complet();
142   Begin
143     self.changeCouleur(clRed);
144     self.complet_:= true;
145     self.Enabled:= false;
146     self.lblNomFilm.Enabled:= false;
147     self.lblHoraire.Enabled:= false;
148   end;
149
150   { ************************************************************************
151     *** Change la couleur du bouton                                      ***
152     *** @params TColor couleur - Couleur du bouton                       ***
153     ************************************************************************ }
154   procedure TImageBouton.changeCouleur(couleur: TColor);
155   Begin
156     with self.Picture do
157     Begin
158       Canvas.Brush.Color:= couleur;
159       Canvas.Rectangle(0, 0, self.Width, self.Height);
160     end;
161   end;
162
163   { ************************************************************************
164     *** Remplace le texte des label avec le text proposer par l'utilisateur  ***
165     *** @params String texte - Valeur du texte                           ***
166     *** @params String objet - label qui est concerné par le changement  ***
167     ***                     (nom -> nom du film, horaire -> horaire dû film) ***
168     ************************************************************************ }
169   procedure TImageBouton.setText(texte: string; objet: string);
```

```
170   Begin
171     if (length(texte) >= MIN_CARACTERE) and (length(texte) <= MAX_CARACTERE) then
172     Begin
173       if objet = 'horaire' then
174         lblHoraire.Caption:= texte;
175
176       if objet = 'nom' then
177         lblNomFilm.Caption:= texte;
178     end;
179   end;
180
181   { ************************************************************************
182     *** Ouvre la fenêtre de réservation                                   ***
183     *** @params TObject Sender - Appel l'objet appeler                    ***
184     ************************************************************************ }
185   procedure TImageBouton.Click(Sender: TObject);
186   Begin
187     if not(self = nil) then
188     Begin
189     if not(self.complet_) or (self.Enabled = false)then
190       FrmOneWayTickets.Reservation(self.lblNomFilm.Caption, self.lblHoraire.caption,
        self.pSalle, self.pSection, self.pPlaces);
191     end;
192   end;
193
194   end.
195
```

```
{ ******************************************************************************
  *** Projet               : OneWayTickets                                     ***
  *** Auteur               : Devaud Alan                                       ***
  *** Fenêtre              : Reservation                                       ***
  *** Description          : Permet d'effectuer une réservation                ***
  *** Version              : 1.0                                               ***
  *** Date de création     : 30.04.2015                                        ***
  ****************************************************************************** }
unit U_OneWayTickets_Reservation;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ComCtrls;

type
  TFrmReservation = class(TForm)
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    lblNomFilm: TLabel;
    lblDuree: TLabel;
    mmoSynopsis: TMemo;
    GroupBox2: TGroupBox;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    EdtNbBilletsEnfants: TEdit;
    UDNBilletsEnfants: TUpDown;
    EdtNbBilletsAdultes: TEdit;
    UDNBilletsAdultes: TUpDown;
    lblPrixEnfants: TLabel;
    lblPrixAdultes: TLabel;
    lblPrixEAA: TLabel;
    EdtNbBilletsEAA: TEdit;
    UDNBilletsEAA: TUpDown;
    lblPrixTotal: TLabel;
    Label8: TLabel;
    GroupBox3: TGroupBox;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    lblSalle: TLabel;
    lblHoraire: TLabel;
    lblPlacesRestantes: TLabel;
    BtnAnnuler: TButton;
    BtnValiderReservation: TButton;
    procedure EdtNbBilletsEAAChange(Sender: TObject);
    procedure UDNBilletsAdultesClick(Sender: TObject; Button: TUDBtnType);
    procedure BtnAnnulerClick(Sender: TObject);
  private
    { Déclarations privées }
  public
    { Déclarations publiques }
    prixTotal : real;
```

```
58      prixEnfant, PrixAdulte, PrixEAA: real;
59      procedure calculTotal();
60    end;
61
62  var
63    FrmReservation: TFrmReservation;
64
65
66  implementation
67
68  {$R *.DFM}
69
70  procedure TFrmReservation.calculTotal();
71  Begin
72    prixTotal:= StrToFloat(EdtNbBilletsEnfants.Text) * prixEnfant;
73    prixTotal:= prixTotal + StrToFloat(EdtNbBilletsAdultes.Text) * PrixAdulte;
74    prixTotal:= prixTotal + StrToFloat(EdtNbBilletsEAA.Text) * PrixEAA;
75
76    lblPrixTotal.Caption:= FloatToStr(prixTotal) + ' CHF';
77  end;
78
79  { ************************************************************************
80    *** Vérifie que les champs on une valeur supérieur à 0                 ***
81    ************************************************************************ }
82  procedure TFrmReservation.EdtNbBilletsEAAChange(Sender: TObject);
83  var
84    places : integer;
85  begin
86    places:= StrToInt(EdtNbBilletsEnfants.Text) + StrToInt(EdtNbBilletsAdultes.Text) +
      StrToInt(EdtNbBilletsEAA.Text);
87
88    if ((StrToInt(EdtNbBilletsEnfants.Text) > 0) or (StrToInt(EdtNbBilletsAdultes.Text)
       > 0) or
89      (StrToInt(EdtNbBilletsEAA.Text) > 0)) and (places <= StrToInt(lblPlacesRestantes.
        Caption) ) then
90      BtnValiderReservation.Enabled:= true
91    else
92      BtnValiderReservation.Enabled:= false;
93
94
95  end;
96
97  { ************************************************************************
98    *** Procedure d'un bouton UpDown (Calcul le prix total)                ***
99    ************************************************************************ }
100 procedure TFrmReservation.UDNBilletsAdultesClick(Sender: TObject;
101   Button: TUDBtnType);
102 begin
103   calculTotal();
104 end;
105
106 { ************************************************************************
107   *** Procedure d'annulation                                            ***
108   ************************************************************************ }
109 procedure TFrmReservation.BtnAnnulerClick(Sender: TObject);
110 var
111   reponse: word;
```

```
112   Begin
113     reponse:= MessageDlg('Etes vous sûr de vouloir annuler la réservation ?',
        mtConfirmation, [mbYes, mbNo, mbCancel], 0);
114
115     if reponse = mrYes then
116       ModalResult:= mrCancel;
117   end;
118
119   end.
120
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                ***
3    *** Fenêtre              : Login                                        ***
4    *** Auteur               : Devaud Alan                                  ***
5    *** Description          : Permet de se connecter en tant qu'admin      ***
6    *** Version              : 1.0                                          ***
7    *** Date de création     : 05.05.2015                                   ***
8    *************************************************************************** }
9  unit U_OneWayTickets_Login;
10
11 interface
12
13 uses
14   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
15   StdCtrls;
16
17 type
18   TFrmLogin = class(TForm)
19     Label1: TLabel;
20     edtMdpAdmin: TEdit;
21     BtnConnexion: TButton;
22     BtnAnnuler: TButton;
23     procedure edtMdpAdminChange(Sender: TObject);
24   private
25     { Déclarations privées }
26   public
27     { Déclarations publiques }
28   end;
29
30 var
31   FrmLogin: TFrmLogin;
32
33 implementation
34
35 {$R *.DFM}
36
37 { ***************************************************************************
38   *** Vérifie si le champ text a du contenu à chaque changement          ***
39   *************************************************************************** }
40 procedure TFrmLogin.edtMdpAdminChange(Sender: TObject);
41 var
42   mdp : string;
43 begin
44   mdp:= edtMdpAdmin.Text;
45
46   if Length(mdp) >= 1 then
47     BtnConnexion.Enabled:= true
48   else
49     BtnConnexion.Enabled:= false;
50 end;
51
52 end.
53
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                ***
3    *** Fenêtre              : Menu administrateur                          ***
4    *** Auteur               : Devaud Alan                                  ***
5    *** Description          : Permet la gestion admin et de visionner les  ***
6    ***                        statistique de vente                         ***
7    *** Version              : 1.0                                          ***
8    *** Date de création     : 05.05.2015                                   ***
9    *************************************************************************** }
10 unit U_OneWayTickets_MenuAdministrateur;
11
12 interface
13
14 uses
15   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
16   StdCtrls, Menus;
17
18 type
19   TFrmMenuAdministrateur = class(TForm)
20     GroupBox1: TGroupBox;
21     Label1: TLabel;
22     Label2: TLabel;
23     Label3: TLabel;
24     Label4: TLabel;
25     Label5: TLabel;
26     lblBilletTotalVendu: TLabel;
27     lblBilletVenduMoisCourrant: TLabel;
28     lblBilletEnfantsVendu: TLabel;
29     lblBilletAdultesVendu: TLabel;
30     lblBilletEtudiantsVendu: TLabel;
31     GroupBox2: TGroupBox;
32     Label6: TLabel;
33     lblFilmPlusVu: TLabel;
34     BtnDeconnexion: TButton;
35     MainMenu1: TMainMenu;
36     Fichier1: TMenuItem;
37     Gestion1: TMenuItem;
38     Dconnexion1: TMenuItem;
39     Films1: TMenuItem;
40     Salles1: TMenuItem;
41     Seance1: TMenuItem;
42     procedure Films1Click(Sender: TObject);
43     procedure Salles1Click(Sender: TObject);
44     procedure Seance1Click(Sender: TObject);
45     procedure BtnDeconnexionClick(Sender: TObject);
46     procedure FormClose(Sender: TObject; var Action: TCloseAction);
47   private
48     { Déclarations privées }
49   public
50     { Déclarations publiques }
51   end;
52
53 var
54   FrmMenuAdministrateur: TFrmMenuAdministrateur;
55
56 const
57   FRM_MODAL_HEIGHT : integer = 359; // Hauteur d'une fenêtre modal
```

```
58
59  implementation
60
61  uses U_OneWayTickets_GestionFilms, U_OneWayTickets_GestionSalles,
62    U_OneWayTickets_GestionSeances, U_OneWayTicket;
63
64  {$R *.DFM}
65
66
67  { ************************************************************************
68    *** Affiche la fenêtre de gestion des films                          ***
69    ************************************************************************ }
70  procedure TFrmMenuAdministrateur.Films1Click(Sender: TObject);
71  begin
72    // Initialise la fenêtre
73    FrmGestionFilms.Height:= FRM_MODAL_HEIGHT;
74    FrmGestionFilms.chargeListeFilm();
75    FrmGestionFilms.LbxListeFilms.ItemIndex:= 0;
76    FrmGestionFilms.Visible:= true;
77  end;
78
79  { ************************************************************************
80    *** Affiche la fenêtre de gestion des salles                         ***
81    ************************************************************************ }
82  procedure TFrmMenuAdministrateur.Salles1Click(Sender: TObject);
83  begin
84    FrmGestionSalles.Height:= FRM_MODAL_HEIGHT; // Initialise la taille de la fenêtre
85    FrmGestionSalles.chargeListeSalles();
86    FrmGestionSalles.LbxListeSalles.ItemIndex:= 0;
87    FrmGestionSalles.Visible:= true;
88  end;
89
90  { ************************************************************************
91    *** Affichage de la gestion des séances                              ***
92    ************************************************************************ }
93  procedure TFrmMenuAdministrateur.Seance1Click(Sender: TObject);
94  begin
95    FrmGestionSeances.Height:= FRM_MODAL_HEIGHT; // Initialise la taille de la fenêtre
96    FrmGestionSeances.chargeListeSeances();
97    FrmGestionSeances.LbxListeSeances.ItemIndex:= 0;
98    FrmGestionSeances.showModal;
99  end;
100
101
102 { ************************************************************************
103   *** Déconnecte l'administrateur pour afficher le menu utilisateur    ***
104   ************************************************************************ }
105 procedure TFrmMenuAdministrateur.BtnDeconnexionClick(Sender: TObject);
106 begin
107   FrmOneWayTickets.Initialisation();
108   FrmMenuAdministrateur.Visible:= false;
109   FrmOneWayTickets.Visible:= true;
110 end;
111
112 { ************************************************************************
113   *** Affiche le menu utilisateur                                      ***
114   ************************************************************************ }
```

```
115   procedure TFrmMenuAdministrateur.FormClose(Sender: TObject;
116     var Action: TCloseAction);
117   begin
118     FrmOneWayTickets.Visible:= true;
119   end;
120
121   end.
122
```

```
1   { ***************************************************************************
2     *** Projet               : OneWayTickets                                ***
3     *** Fenêtre              : GestionFilms                                 ***
4     *** Auteur               : Devaud Alan                                  ***
5     *** Description          : Donne accès au fonctionnalité de la gestion  ***
6     ***                       des films                                     ***
7     *** Version              : 1.0                                          ***
8     *** Date de création     : 05.05.2015                                   ***
9     ***         Modification :                                              ***
10    ***           06.05.2015 - AD - Modification de la modification        ***
11    ***                              Ajout de la suppression des films      ***
12    *************************************************************************** }
13  unit U_OneWayTickets_GestionFilms;
14
15  interface
16
17  uses
18    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
19    StdCtrls, Grids, ComCtrls, U_FP;
20
21  type
22    TFrmGestionFilms = class(TForm)
23      Label1: TLabel;
24      LbxListeFilms: TListBox;
25      BtnModifier: TButton;
26      BtnSupprimer: TButton;
27      BtnAjouter: TButton;
28      GroupBox1: TGroupBox;
29      Label2: TLabel;
30      edtNomFilm: TEdit;
31      edtDureeFilm: TEdit;
32      Label3: TLabel;
33      Label4: TLabel;
34      MmoSynopsis: TMemo;
35      BtnValiderModification: TButton;
36      BtnAnuuler: TButton;
37      edtPrixFilm: TEdit;
38      Label5: TLabel;
39      UDDuree: TUpDown;
40      UDPrix: TUpDown;
41      Label6: TLabel;
42      Label7: TLabel;
43      Label8: TLabel;
44      Label9: TLabel;
45      procedure BtnModifierClick(Sender: TObject);
46      procedure BtnAnuulerClick(Sender: TObject);
47      procedure BtnAjouterClick(Sender: TObject);
48      procedure edtDureeFilmChange(Sender: TObject);
49      procedure edtDureeFilmKeyPress(Sender: TObject; var Key: Char);
50      procedure testEdit();
51      procedure MmoSynopsisChange(Sender: TObject);
52      procedure BtnValiderModificationClick(Sender: TObject);
53      procedure changeEtat(active: boolean);
54      procedure BtnSupprimerClick(Sender: TObject);
55    private
56      { Déclarations privées }
57    public
```

```
58      { Déclarations publiques }
59      procedure chargeListeFilm(); // Est disponible depuis l'extérieur
60    end;
61
62  var
63    FrmGestionFilms      : TFrmGestionFilms;
64    listeFilms           : array of TFilm; // Enregistre tous les films pour une
      modification future
65
66  const
67    FRM_HEIGHT_MAX : integer = 584;
68    FRM_HEIGHT_MIN : integer = 358;
69
70  implementation
71
72  uses U_OneWayTickets_AjouterFilm, U_OneWayTicket;
73
74  {$R *.DFM}
75
76  { ***************************************************************************
77    *** Change l'état des éléments                                          ***
78    *** @params Boolean active - true ou false                              ***
79    *************************************************************************** }
80  procedure TFrmGestionFilms.changeEtat(active: boolean);
81  Begin
82    LbxListeFilms.Enabled:= active;
83    BtnModifier.Enabled:= active;
84    BtnAjouter.Enabled:= active;
85    BtnSupprimer.Enabled:= active;
86  end;
87
88  { ***************************************************************************
89    *** Permet de tester si les champs sont remplis                         ***
90    *************************************************************************** }
91  procedure TFrmGestionFilms.testEdit();
92  Begin
93    if (Length(edtNomFilm.Text) > 0) and (Length(edtDureeFilm.Text) > 0) and
94      (Length(edtPrixFilm.Text) > 0) and (Length(MmoSynopsis.Text) > 0) then
95      BtnValiderModification.Enabled:= true
96    else
97      BtnValiderModification.Enabled:= false;
98  end;
99
100 { ***************************************************************************
101   *** Charge et affiche les films depuis un fichier csv                   ***
102   *************************************************************************** }
103 procedure TFrmGestionFilms.chargeListeFilm();
104 var
105   ligneFormate : String;
106   OutPutList : TStringList;
107   valeur: TValeur;
108   i, nbLigne: integer;
109 Begin
110   LbxListeFilms.Clear;
111   nbLigne:= nbLignesFichier(FICHIER_FILMS);
112   SetLength(listeFilms, nbLigne);
113
```

```
114     // Charge les valeurs du ficheir salles
115     valeur:= lireFichier(FICHIER_FILMS);
116     for i:= 1 to length(valeur) - 1 do
117     Begin
118       // Test s'il y a une valeur
119       if valeur[i] = '' then
120         break;
121
122       // Split la ligne
123       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
124
125       // Formate le text pour l'affichage
126       ligneFormate:= ajusterText(OutPutList[0], 15) + AjoutEspace(5);
127       ligneFormate:= ligneFormate + AjoutEspace(10) + OutPutList[1] + AjoutEspace(10);
128       ligneFormate:= ligneFormate + AjoutEspace(10) + ajusterText(OutPutList[2], 32) +
          AjoutEspace(10);
129       ligneFormate:= ligneFormate + AjoutEspace(10) + OutPutList[3];
130       LbxListeFilms.Items.Add(ligneFormate);
131
132       //Met les données dans la liste
133       listeFilms[i - 1].NomFilm:= OutPutList[0];
134       listeFilms[i - 1].Duree:= OutPutList[1];
135       listeFilms[i - 1].Synopsis:= OutPutList[2];
136       listeFilms[i - 1].Prix:= OutPutList[3];
137     end;
138   end;
139
140   { ************************************************************************
141     *** Ouvre les paramètres de modifications                            ***
142     ************************************************************************ }
143   procedure TFrmGestionFilms.BtnModifierClick(Sender: TObject);
144   var
145     index : integer;
146   begin
147     // Récupère l'indice
148     index:= LbxListeFilms.ItemIndex;
149
150     // Affiche les données dans les bon champs
151     edtNomFilm.Text:= listefilms[index].NomFilm;
152     edtDureeFilm.Text:= listefilms[index].Duree;
153     edtPrixFilm.Text:= listeFilms[index].Prix;
154     MmoSynopsis.Text:= listefilms[index].Synopsis;
155
156     changeEtat(false);
157     self.Height:= FRM_HEIGHT_MAX;
158   end;
159
160   { ************************************************************************
161     *** Annule les modification                                          ***
162     ************************************************************************ }
163   procedure TFrmGestionFilms.BtnAnuulerClick(Sender: TObject);
164   var
165     reponse : word;
166   begin
167     // Récupère la réponse
168     reponse:= MessageDlg('Etes-vous sûr de vouloir annuler les modifications ?',
      mtConfirmation, [mbYes, mbNo, mbCancel], 0);
```

```
169
170     // Test si la réponse est oui
171     if reponse = mrYes then
172     Begin
173       changeEtat(true);
174       self.Height:= FRM_HEIGHT_MIN;
175     end;
176
177   end;
178
179   { *************************************************************************
180     *** Ouvre la fenêtre pour ajouter un film                             ***
181     ************************************************************************* }
182   procedure TFrmGestionFilms.BtnAjouterClick(Sender: TObject);
183   var
184     valeurs: array [0..3] of String;
185   begin
186     // Initialisation des composants
187     with FrmAjouterFilm do
188     Begin
189       edtNomFilm.Text:= '';
190       edtDureeFilm.Text:= '90';
191       edtPrixFilm.Text:= '5';
192       MmoSynopsis.Clear;
193     end;
194
195     if FrmAjouterFilm.showModal = mrOk then
196     Begin
197
198       // Initialise les valeurs
199       valeurs[0]:= FrmAjouterFilm.edtNomFilm.Text;
200       valeurs[1]:= FrmAjouterFilm.edtDureeFilm.Text;
201       valeurs[2]:= FrmAjouterFilm.MmoSynopsis.Text;
202       valeurs[3]:= FrmAjouterFilm.edtPrixFilm.Text;
203
204       // Test si le film exist déjà
205       if not(valeurExists(FICHIER_FILMS, valeurs[0])) then
206       Begin
207         if ajoutUneLigne(valeurs, FICHIER_FILMS) then
208           MessageDlg('Film ajouté avec succès !', mtInformation, [mbOk, mbCancel], 0)
209         else
210           MessageDlg('Une erreur est survenue lors de l''ajout !', mtError, [mbOk,
              mbCancel], 0);
211       end
212       else
213       Begin
214         MessageDlg('Ce film exist déjà !', mtError, [mbOk, mbCancel], 0);
215       end;
216
217       chargeListeFilm();
218     end;
219   end;
220
221   { *************************************************************************
222     *** Test de la valeur des champs text                                 ***
223     ************************************************************************* }
224   procedure TFrmGestionFilms.edtDureeFilmChange(Sender: TObject);
```

```
225   begin
226     if ((Sender as TEdit).Name = 'edtDureeFilm') or ((Sender as TEdit).Name =
      'edtPrixFilm') then
227       if StrToInt((Sender as TEdit).Text) > 400 then
228         (Sender as TEdit).Text:= '10';
229
230     testEdit();
231   end;
232
233   { ***************************************************************************
234     *** Filtre les caractères d'un champ text                                 ***
235     *************************************************************************** }
236   procedure TFrmGestionFilms.edtDureeFilmKeyPress(Sender: TObject;
237     var Key: Char);
238   begin
239     if not(key in['0'..'9']) then
240       key:= #0;
241   end;
242
243   { ***************************************************************************
244     *** Procédure lors d'un changement dans le memo                           ***
245     *************************************************************************** }
246   procedure TFrmGestionFilms.MmoSynopsisChange(Sender: TObject);
247   begin
248     testEdit();
249   end;
250
251   { ***************************************************************************
252     *** Procedure de la validation de la modification                         ***
253     *************************************************************************** }
254   procedure TFrmGestionFilms.BtnValiderModificationClick(Sender: TObject);
255   var
256     index, i: integer;
257     valeurs: TValeurs;
258     reponse: word;
259   begin
260     reponse:= 0;
261     index:= LbxListeFilms.ItemIndex;
262
263     if not(valeurExists(FICHIER_FILMS, edtNomFilm.Text)) then
264     Begin
265       setLength(valeurs, length(listeFilms) + 1, 4);
266
267       // Récupère les informations du film modifier et les mets dans le tableau
268       listeFilms[index].NomFilm:= edtNomFilm.Text;
269       listeFilms[index].Duree:= edtDureeFilm.Text;
270       listeFilms[index].Prix:= edtPrixFilm.Text;
271       listeFilms[index].Synopsis:= MmoSynopsis.Text;
272
273       // Initialise le tableau pour écrire dans le fichier
274       valeurs[0][0]:= 'NomFilm';
275       valeurs[0][1]:= 'Duree';
276       valeurs[0][2]:= 'Synopsis';
277       valeurs[0][3]:= 'Prix';
278
279       for i:= 0 to length(listeFilms) - 1 do
280       Begin
```

```
281         valeurs[i + 1][0]:= listeFilms[i].NomFilm;
282         valeurs[i + 1][1]:= listeFilms[i].Duree;
283         valeurs[i + 1][2]:= listeFilms[i].Synopsis;
284         valeurs[i + 1][3]:= listeFilms[i].Prix;
285       end;
286
287
288       if ecritDansFichier(valeurs, FICHIER_FILMS) then
289         MessageDlg('Modification effectuée avec succès !', mtInformation, [mbOk,
          mbCancel], 0)
290       else
291         MessageDlg('Une erreur est survenue lors de la modification !', mtError, [mbOk,
           mbCancel], 0);
292
293       chargeListeFilm(); // Charge la nouvelle liste
294     end
295     else
296     Begin
297       reponse:= MessageDlg('Ce nom de film est déjà prit !', mtError ,[mbOk, mbRetry,
        mbCancel], 0);
298     end;
299
300     if not(reponse = mrRetry) then
301     Begin
302       // Remet la fenêtre comme elle se trouve au départ
303       changeEtat(true);
304       LbxListeFilms.ItemIndex:= 0;
305       self.Height:= FRM_HEIGHT_MIN;
306     end;
307
308     LbxListeFilms.ItemIndex:= index;
309   end;
310
311   { ************************************************************************
312     *** Procedure de la suppression                                      ***
313     ************************************************************************ }
314   procedure TFrmGestionFilms.BtnSupprimerClick(Sender: TObject);
315   var
316     index, i: integer;
317     valeurs: TValeurs;
318     reponse : word;
319   begin
320     // Demande confirmation
321     reponse:= MessageDlg('Voulez-vous vraiment supprimer ce film ?', mtWarning, [mbYes,
      mbNo], 0);
322
323     // Test la réponse de la boîte de dialogue
324     if reponse = mrYes then
325     Begin
326       // Initialisation des variables
327       setLength(valeurs, length(listeFilms) + 1, 4);
328       index:= LbxListeFilms.ItemIndex;
329
330       // "Suppresion" du film
331       listeFilms[index].NomFilm:= '';
332       listeFilms[index].Duree:= '';
333       listeFilms[index].Prix:= '';
```

```
334       listeFilms[index].Synopsis:= '';
335
336       index:= 0;
337       valeurs[0][0]:= 'NomSalle';
338       valeurs[0][1]:= 'Places';
339
340       // Nettoyage du tableau (Enlève l'élément vide)
341       for i:= 0 to length(listeFilms) - 1 do
342       Begin
343         if listeFilms[i].NomFilm <> '' then
344         Begin
345           valeurs[index + 1][0]:= listeFilms[i].NomFilm;
346           valeurs[index + 1][1]:= listeFilms[i].Duree;
347           valeurs[index + 1][2]:= listeFilms[i].Synopsis;
348           valeurs[index + 1][3]:= listeFilms[i].Prix;
349           inc(index);
350         end;
351       end;
352
353       if ecritDansFichier(valeurs, FICHIER_FILMS) then
354         MessageDlg('Le film a bien été supprimé !', mtInformation, [mbOk, mbCancel], 0)
355       else
356         MessageDlg('Une erreur est survenue lors de la suppression !', mtError, [mbOk,
          mbCancel], 0);
357
358       chargeListeFilm();
359     end;
360   end;
361
362   end.
363
```

```
1   unit U_OneWayTickets_AjouterFilm;
2
3   interface
4
5   uses
6     Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
7     StdCtrls, ComCtrls;
8
9   type
10    TFrmAjouterFilm = class(TForm)
11      Label1: TLabel;
12      edtNomFilm: TEdit;
13      Label2: TLabel;
14      edtDureeFilm: TEdit;
15      Label3: TLabel;
16      MmoSynopsis: TMemo;
17      BtnAnnuler: TButton;
18      BtnAjouter: TButton;
19      Label4: TLabel;
20      edtPrixFilm: TEdit;
21      UDDuree: TUpDown;
22      UDPrix: TUpDown;
23      procedure edtDureeFilmChange(Sender: TObject);
24      procedure edtDureeFilmKeyPress(Sender: TObject; var Key: Char);
25      procedure testEdit();
26      procedure MmoSynopsisChange(Sender: TObject);
27      procedure BtnAnnulerClick(Sender: TObject);
28    private
29      { Déclarations privées }
30    public
31      { Déclarations publiques }
32    end;
33
34  var
35    FrmAjouterFilm: TFrmAjouterFilm;
36
37  implementation
38
39  {$R *.DFM}
40
41  { ************************************************************************
42    *** Permet de tester si les champs sont remplis                      ***
43    ************************************************************************ }
44  procedure TFrmAjouterFilm.testEdit();
45  Begin
46    if (Length(edtNomFilm.Text) > 0) and (Length(edtDureeFilm.Text) > 0) and
47      (Length(edtPrixFilm.Text) > 0) and (Length(MmoSynopsis.Text) > 0) then
48      BtnAjouter.Enabled:= true
49    else
50      BtnAjouter.Enabled:= false;
51  end;
52
53  { ************************************************************************
54    *** Test de la valeur d'un champ text                                ***
55    ************************************************************************ }
56  procedure TFrmAjouterFilm.edtDureeFilmChange(Sender: TObject);
57  begin
```

```
58    if ((Sender as TEdit).Name = 'edtDureeFilm') or ((Sender as TEdit).Name =
      'edtPrixFilm') then
59      if StrToInt((Sender as TEdit).Text) > 400 then
60        (Sender as TEdit).Text:= '10';
61
62    testEdit();
63  end;
64
65  { *************************************************************************
66    *** Filtre les caractères d'un champ text                             ***
67    ************************************************************************* }
68  procedure TFrmAjouterFilm.edtDureeFilmKeyPress(Sender: TObject;
69    var Key: Char);
70  begin
71    if not(key in['0'..'9']) then
72      key:= #0;
73  end;
74
75
76  { *************************************************************************
77    *** Procédure lors d'un changement dans le memo                       ***
78    ************************************************************************* }
79  procedure TFrmAjouterFilm.MmoSynopsisChange(Sender: TObject);
80  begin
81    testEdit();
82  end;
83
84  { *************************************************************************
85    *** Procedure d'annulation                                            ***
86    ************************************************************************* }
87  procedure TFrmAjouterFilm.BtnAnnulerClick(Sender: TObject);
88  var
89    reponse : word;
90  begin
91    reponse:= MessageDlg('Etes vous sûr de vouloir annuler l''ajout ?', mtConfirmation,
       [mbYes, mbNo, mbCancel], 0);
92
93    if reponse = mrYes then
94      ModalResult:= mrCancel;
95  end;
96
97  end.
98
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                ***
3    *** Fenêtre              : GestionSalles                                ***
4    *** Auteur               : Devaud Alan                                  ***
5    *** Description          : Donne accès au fonctionnalité de la gestion  ***
6    ***                        des salles                                   ***
7    *** Version              : 1.0                                          ***
8    *** Date de création     : 05.05.2015                                   ***
9    ***         Modification :                                              ***
10   ***           06.05.2015 - AD - Ajout de l'ajout d'une salle           ***
11   ***                             Ajout de la modification d'une salle   ***
12   ***                             Ajout de la suppression d'une salle    ***
13   ***                             Ajout de la sécurité et de l'affichage ***
14   *************************************************************************** }
15 unit U_OneWayTickets_GestionSalles;
16
17 interface
18
19 uses
20   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
21   StdCtrls, ComCtrls, U_FP;
22
23 type
24   TFrmGestionSalles = class(TForm)
25     Label2: TLabel;
26     LbxListeSalles: TListBox;
27     BtnModifier: TButton;
28     BtnSupprimer: TButton;
29     BtnAjouter: TButton;
30     GroupBox1: TGroupBox;
31     Label1: TLabel;
32     Label3: TLabel;
33     edtNomSalle: TEdit;
34     edtPlacesMax: TEdit;
35     BtnValiderModification: TButton;
36     BtnAnuuler: TButton;
37     UDPlacesMax: TUpDown;
38     Label7: TLabel;
39     Label6: TLabel;
40     procedure BtnAnuulerClick(Sender: TObject);
41     procedure BtnModifierClick(Sender: TObject);
42     procedure BtnAjouterClick(Sender: TObject);
43     procedure testEdit();
44     procedure edtNomSalleChange(Sender: TObject);
45     procedure edtPlacesMaxKeyPress(Sender: TObject; var Key: Char);
46     procedure changeEtat(active: boolean);
47     procedure BtnValiderModificationClick(Sender: TObject);
48     procedure BtnSupprimerClick(Sender: TObject);
49   private
50     { Déclarations privées }
51   public
52     { Déclarations publiques }
53     procedure chargeListeSalles();
54   end;
55
56
57
```

```
58
59
60  var
61    FrmGestionSalles : TFrmGestionSalles;
62    listeSalles      : array of TSalle;
63
64  const
65    FRM_HEIGHT_MAX : integer = 487;
66    FRM_HEIGHT_MIN : integer = 358;
67
68  implementation
69
70  uses U_OneWayTickets_AjouterSalle, U_OneWayTicket;
71
72  {$R *.DFM}
73
74
75  { ************************************************************************
76    *** Change l'état des éléments                                       ***
77    *** @params Boolean active - true ou false                           ***
78    ************************************************************************ }
79  procedure TFrmGestionSalles.changeEtat(active: boolean);
80  Begin
81    LbxListeSalles.Enabled:= active;
82    BtnModifier.Enabled:= active;
83    BtnAjouter.Enabled:= active;
84    BtnSupprimer.Enabled:= active;
85  end;
86
87  { ************************************************************************
88    *** Permet de tester si les champs sont remplis                      ***
89    ************************************************************************ }
90  procedure TFrmGestionSalles.testEdit();
91  Begin
92    if (Length(edtNomSalle.Text) > 0) and (Length(edtPlacesMax.Text) > 0) then
93      BtnValiderModification.Enabled:= true
94    else
95      BtnValiderModification.Enabled:= false;
96  end;
97
98  { ************************************************************************
99    *** Charge et affiche les salles depuis un fichier csv              ***
100   ************************************************************************ }
101 procedure TFrmGestionSalles.chargeListeSalles();
102 var
103   ligneFormate : String;
104   OutPutList : TStringList;
105   valeur: TValeur;
106   nbLigne: integer;
107   i: integer;
108 Begin
109   LbxListeSalles.Clear;
110   nbLigne:= nbLignesFichier(FICHIER_SALLES);
111   SetLength(listeSalles, nbLigne);
112
113   // Charge les valeurs du ficheir salles
114   valeur:= lireFichier(FICHIER_SALLES);
```

```
115     for i:= 1 to length(valeur) - 1 do
116     Begin
117       // Test s'il y a une valeur
118       if valeur[i] = '' then
119         break;
120
121       // Split la ligne
122       OutPutList:= Split(valeur[i], CARAC_SEPARATION);
123
124       // Formate le text pour l'affichage
125       ligneFormate:= ajusterText(OutPutList[0], 20) + AjoutEspace(28);
126       ligneFormate:= ligneFormate + AjoutEspace(38) + OutPutList[1];
127       LbxListeSalles.Items.Add(ligneFormate);
128
129       //Met les données dans la liste
130       listeSalles[i - 1].NomSalle:= OutPutList[0];
131       listeSalles[i - 1].Places:= OutPutList[1];
132     end;
133
134   end;
135
136   { ************************************************************************
137     *** Annule les modification                                          ***
138     ************************************************************************ }
139   procedure TFrmGestionSalles.BtnAnuulerClick(Sender: TObject);
140   var
141     reponse : word;
142   begin
143     // Récupère la réponse
144     reponse:= MessageDlg('Etes-vous sûr de vouloir annuler les modifications ?',
      mtConfirmation, [mbYes, mbNo, mbCancel], 0);
145
146     // Test si la réponse est oui
147     if reponse = mrYes then
148     Begin
149       changeEtat(true);
150       self.Height:= FRM_HEIGHT_MIN;
151     end;
152   end;
153
154   { ************************************************************************
155     *** Ouvre les paramètres de modifications                            ***
156     ************************************************************************ }
157   procedure TFrmGestionSalles.BtnModifierClick(Sender: TObject);
158   var
159     index: integer;
160   begin
161     // Récupère l'indice
162     index:= LbxListeSalles.ItemIndex;
163
164     // Affiche les données dans les bon champs
165     edtNomSalle.Text:= listeSalles[index].NomSalle;
166     edtPlacesMax.Text:= listeSalles[index].Places;
167
168     changeEtat(false);
169     self.Height:= FRM_HEIGHT_MAX;
170   end;
```

```
171
172 { ***************************************************************************
173   *** Ouvre la fenêtre pour ajouter une salles                             ***
174   *************************************************************************** }
175 procedure TFrmGestionSalles.BtnAjouterClick(Sender: TObject);
176 var
177   valeurs: array [0..1] of string;
178 begin
179   with FrmAjouterSalle do
180   Begin
181     edtNomSalle.Text:= '';
182     edtPlacesMax.Text:= '1';
183   end;
184
185   if FrmAjouterSalle.showModal = mrOk then
186   Begin
187     // Initialise les valeurs
188     valeurs[0]:= FrmAjouterSalle.edtNomSalle.Text;
189     valeurs[1]:= FrmAjouterSalle.edtPlacesMax.Text;
190
191     // Test si la salle existe déjà
192     if not(valeurExists(FICHIER_SALLES, valeurs[0])) then
193     Begin
194       if ajoutUneLigne(valeurs, FICHIER_SALLES) then
195         MessageDlg('Salle ajoutée avec succès !', mtInformation, [mbOk, mbCancel], 0)
196       else
197         MessageDlg('Une erreur est survenue lors de l''ajout !', mtError, [mbOk,
          mbCancel], 0);
198     end
199     else
200     Begin
201       MessageDlg('La salle existe déjà !', mtError, [mbOk, mbCancel], 0);
202     end;
203
204     chargeListeSalles();
205   end;
206 end;
207
208 { ***************************************************************************
209   *** Procédure lors d'un changement dans un edit                          ***
210   *************************************************************************** }
211 procedure TFrmGestionSalles.edtNomSalleChange(Sender: TObject);
212 begin
213   if (Sender as TEdit).Name = 'edtPlacesMax' then
214     if StrToInt((Sender as TEdit).Text) > 500 then
215       (Sender as TEdit).Text:= '50';
216
217   testEdit();
218 end;
219
220 { ***************************************************************************
221   *** Filtre les caractères d'un champ text                                ***
222   *************************************************************************** }
223 procedure TFrmGestionSalles.edtPlacesMaxKeyPress(Sender: TObject;
224   var Key: Char);
225 begin
226   if not(key in['0'..'9']) then
```

```
227       key:= #0;
228   end;
229
230   { ***************************************************************************
231     *** Procedure de la validation de la modification                        ***
232     *************************************************************************** }
233   procedure TFrmGestionSalles.BtnValiderModificationClick(Sender: TObject);
234   var
235     index, i: integer;
236     valeurs: TValeurs;
237     reponse : word;
238   begin
239     reponse:= 0 ;
240     index:= LbxListeSalles.ItemIndex;
241
242     if not(valeurExists(FICHIER_SALLES, edtNomSalle.Text)) then
243     Begin
244       setLength(valeurs, length(listeSalles) + 1, 2);
245
246       // Récupère les informations du film modifier et les mets dans le tableau
247       listeSalles[index].NomSalle:= edtNomSalle.Text;
248       listeSalles[index].Places:= edtPlacesMax.Text;
249
250       // Initialise le tableau pour écrire dans le fichier
251       valeurs[0][0]:= 'NomSalle';
252       valeurs[0][1]:= 'Places';
253       for i:= 0 to length(listeSalles) - 1 do
254       Begin
255         valeurs[i + 1][0]:= listeSalles[i].NomSalle;
256         valeurs[i + 1][1]:= listeSalles[i].Places;
257       end;
258
259       if ecritDansFichier(valeurs, FICHIER_SALLES) then
260         MessageDlg('Modification effectuée avec succès !', mtInformation, [mbOk,
          mbCancel], 0)
261       else
262         MessageDlg('Une erreur est survenue lors de la modificatgion !', mtError, [mbOk
          , mbCancel], 0);
263
264       chargeListeSalles(); // Charge la nouvelle liste
265     end
266     else
267     Begin
268       reponse:= MessageDlg('Ce nom de salle est déjà prit !', mtError ,[mbOk, mbRetry,
          mbCancel], 0);
269     end;
270
271     if not(reponse = mrRetry) then
272     Begin
273       // Remet la fenêtre comme elle se trouve au départ
274       changeEtat(true);
275       self.Height:= FRM_HEIGHT_MIN;
276     end;
277
278     LbxListeSalles.ItemIndex:= index;
279
280   end;
```

```
281
282 { ****************************************************************************
283   *** Procedure de la suppression                                          ***
284   **************************************************************************** }
285 procedure TFrmGestionSalles.BtnSupprimerClick(Sender: TObject);
286 var
287   index, i: integer;
288   valeurs: TValeurs;
289   reponse : word;
290 begin
291   // Demande confirmation
292   reponse:= MessageDlg('Voulez-vous vraiment supprimer cette salle ?', mtWarning, [
      mbYes,mbNo], 0);
293
294   // Test la réponse de la boîte de dialogue
295   if reponse = mrYes then
296   Begin
297     // Initialisation des variables
298     setLength(valeurs, length(listeSalles) + 1, 2);
299     index:= LbxListeSalles.ItemIndex;
300
301     // "Suppresion" du film
302     listeSalles[index].NomSalle:= '';
303     listeSalles[index].Places:= '';
304
305     index:= 0;
306     valeurs[0][0]:= 'NomSalle';
307     valeurs[0][1]:= 'Places';
308
309     // Nettoyage du tableau (Enlève l'élément vide)
310     for i:= 0 to length(listeSalles) - 1 do
311     Begin
312       if listeSalles[i].NomSalle <> '' then
313       Begin
314         valeurs[index + 1][0]:= listeSalles[i].NomSalle;
315         valeurs[index + 1][1]:= listeSalles[i].Places;
316         inc(index);
317       end;
318     end;
319
320     if ecritDansFichier(valeurs, FICHIER_SALLES) then
321       MessageDlg('La salle a bien étée supprimée !', mtInformation, [mbOk, mbCancel],
        0)
322     else
323       MessageDlg('Une erreur est survenue lors de la suppression !', mtError, [mbOk,
        mbCancel], 0);
324
325     chargeListeSalles();
326   end;
327 end;
328
329 end.
330
```

```
1   unit U_OneWayTickets_AjouterSalle;
2
3   interface
4
5   uses
6     Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
7     StdCtrls, ComCtrls;
8
9   type
10    TFrmAjouterSalle = class(TForm)
11      Label1: TLabel;
12      edtNomSalle: TEdit;
13      Label2: TLabel;
14      edtPlacesMax: TEdit;
15      BtnAnnuler: TButton;
16      BtnAjouter: TButton;
17      UDPlacesMax: TUpDown;
18      procedure edtNomSalleChange(Sender: TObject);
19      procedure edtPlacesMaxKeyPress(Sender: TObject; var Key: Char);
20      procedure testEdit();
21      procedure BtnAnnulerClick(Sender: TObject);
22    private
23      { Déclarations privées }
24    public
25      { Déclarations publiques }
26    end;
27
28  var
29    FrmAjouterSalle: TFrmAjouterSalle;
30
31  implementation
32
33  {$R *.DFM}
34
35  { ***************************************************************************
36    *** Permet de tester si les champs sont remplis                         ***
37    *************************************************************************** }
38  procedure TFrmAjouterSalle.testEdit();
39  Begin
40    if (Length(edtNomSalle.Text) > 0) and (Length(edtPlacesMax.Text) > 0) then
41      BtnAjouter.Enabled:= true
42    else
43      BtnAjouter.Enabled:= false;
44  end;
45
46  { ***************************************************************************
47    *** Procédure lors d'un changement dans un edit                         ***
48    *************************************************************************** }
49  procedure TFrmAjouterSalle.edtNomSalleChange(Sender: TObject);
50  begin
51    if (Sender as TEdit).Name = 'edtPlacesMax' then
52      if StrToInt((Sender as TEdit).Text) > 500 then
53        (Sender as TEdit).Text:= '50';
54
55    testEdit();
56  end;
57
```

```
58  { ************************************************************************
59    *** Filtre les caractères d'un champ text                             ***
60    ************************************************************************ }
61  procedure TFrmAjouterSalle.edtPlacesMaxKeyPress(Sender: TObject;
62    var Key: Char);
63  begin
64    if not(key in['0'..'9']) then
65      key:= #0;
66  end;
67
68  { ************************************************************************
69    *** Procedure d'annulation                                            ***
70    ************************************************************************ }
71  procedure TFrmAjouterSalle.BtnAnnulerClick(Sender: TObject);
72  var
73    reponse : word;
74  begin
75    reponse:= MessageDlg('Etes vous sûr de vouloir annuler l''ajout ?', mtConfirmation,
       [mbYes, mbNo, mbCancel], 0);
76
77    if reponse = mrYes then
78      ModalResult:= mrCancel;
79  end;
80
81  end.
82
```

```
{ ******************************************************************************
  *** Projet               : OneWayTickets                                   ***
  *** Fenêtre              : GestionSeances                                  ***
  *** Auteur               : Devaud Alan                                     ***
  *** Description          : Donne accès au fonctionnalité de la gestion     ***
  ***                         des séances                                    ***
  *** Version              : 1.0                                             ***
  *** Date de création     : 05.05.2015                                      ***
  ***         Modification :                                                 ***
  ***           06.05.2015 - AD - Ajout de l'ajout de séances               ***
  ***                             Ajout de la modification de séances        ***
  ***           07.05.2015 - AD - Modificatoin de la modification           ***
  ***                             Ajout de la supression de séances          ***
  ***                             Ajout d'éléments de sécurité               ***
  ****************************************************************************** }
unit U_OneWayTickets_GestionSeances;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, IniFiles, U_FP;

type
  TFrmGestionSeances = class(TForm)
    Label1: TLabel;
    LbxListeSeances: TListBox;
    BtnModifier: TButton;
    BtnSupprimer: TButton;
    BtnAjouter: TButton;
    GroupBox1: TGroupBox;
    Label2: TLabel;
    BtnValiderModification: TButton;
    BtnAnuuler: TButton;
    Label3: TLabel;
    CbxFilm: TComboBox;
    CbxSalle: TComboBox;
    GroupBox2: TGroupBox;
    ChxLundi: TCheckBox;
    ChxSamedi: TCheckBox;
    ChxVendredi: TCheckBox;
    ChxDimanche: TCheckBox;
    ChxMardi: TCheckBox;
    ChxMercredi: TCheckBox;
    ChxJeudi: TCheckBox;
    GroupBox3: TGroupBox;
    Label4: TLabel;
    edtHeure1: TEdit;
    edtHeure2: TEdit;
    Label5: TLabel;
    edtHeure3: TEdit;
    Label6: TLabel;
    edtHeure4: TEdit;
    Label7: TLabel;
    procedure BtnModifierClick(Sender: TObject);
    procedure BtnAnuulerClick(Sender: TObject);
```

```
57        procedure BtnAjouterClick(Sender: TObject);
58        procedure changeEtat(active: boolean);
59        procedure BtnValiderModificationClick(Sender: TObject);
60        procedure BtnSupprimerClick(Sender: TObject);
61        procedure edtHeure1Change(Sender: TObject);
62        procedure edtHeure1KeyPress(Sender: TObject; var Key: Char);
63        procedure edtHeure1KeyUp(Sender: TObject; var Key: Word;
64          Shift: TShiftState);
65      private
66        { Déclarations privées }
67      public
68        { Déclarations publiques }
69        procedure chargeListeSeances();
70      end;
71
72    // Tableau pour les clés et les valeurs des champs pour le fichier INI
73    Type
74      TCleVal = array [0..7] of array [0..1] of string;
75
76    var
77      FrmGestionSeances : TFrmGestionSeances;
78      ListeSeances      :  array of TSeance;
79
80    const
81      FRM_WIDTH_MAX             : integer = 580;
82      FRM_WIDTH_MIN             : integer = 359;
83      MAX_TABLEAU_CLEVAL_INDICE : integer = 7;  // Taille maximum de la première
      dimension du tableau
84      HEURE_SEPARATEUR          : String  = ':';
85      MAX_HEURES                : integer = 23;
86      MAX_MINUTES               : integer = 59;
87
88    implementation
89
90    uses U_OneWayTickets_AjouterSeance, U_OneWayTicket;
91
92    {$R *.DFM}
93
94    { ************************************************************************
95      *** Enregistre dans un fichier ini                                   ***
96      *** @params String fichier - Chemin et nom du fichier INI            ***
97      *** @params String parametre - Nom du paramètre dans le fichier INI  ***
98      *** @params TCleVal ValCle - Clé et valeur à entrer dans le fichier INI  ***
99      *** @result boolean - Renvoi vrai si tout c'est bien passé           ***
100     ************************************************************************ }
101   function sauvegardeIni(fichier: string; parametre: string; ValCle: TCleVal): boolean;
102   var
103     fichierIni: TInifile;
104     i: integer;
105     sauvegarde: boolean;
106   Begin
107     sauvegarde:= false;
108
109     if FileExists(fichier) then
110     Begin
111       fichierIni:= TIniFile.Create(fichier);  // Initialise le fichier INI
112
```

```
113       // Parcoure le tableau et ajoute les éléments
114       for i:= 0 to MAX_TABLEAU_CLEVAL_INDICE do
115       begin
116         fichierIni.WriteString(parametre, ValCle[i][0], ValCle[i][1]);
117       end;
118
119       fichierIni.free;  // Libère le fichier
120
121       sauvegarde:= true;
122     end;
123
124     Result:= sauvegarde;
125   end;
126
127   { ************************************************************************
128     *** Compte le nombre de section existant dans un fichier ini          ***
129     *** @params String fichier - Chemin et nom du fichier INI             ***
130     *** @result Integer - Nombre de section                               ***
131     ************************************************************************ }
132   function compteSectionIni(fichier: string): integer;
133   var
134     FichierIni: TIniFile;
135     Section: TStringList;
136     NbSection: integer;
137   Begin
138     NbSection:= 0;
139
140     if FileExists(fichier) then
141     Begin
142       FichierIni:= TIniFile.Create(fichier);
143       Section:= TStringList.Create;
144       FichierIni.ReadSections(Section); // Récupère toutes les sections
145       NbSection:= Section.Count; // Compte le nombre de sections
146       Section.Free;
147       FichierIni.Free;
148     end;
149
150     Result:= NbSection;
151   end;
152
153   { ************************************************************************
154     *** Passe les jours en valeur numérique                               ***
155     *** @params TList chxList - Listes des checkbox                       ***
156     *** @Result string - Chaine comportant les jours                      ***
157     ************************************************************************ }
158   function jourEnNombre(chxList: TList): string;
159   var
160     chx: TCheckBox;
161     i: integer;
162     valeurs: string;
163   Begin
164     // Initialisation
165     valeurs:= '';
166
167     // Parcoure la liste des checkbox
168     for i:= 1 to chxList.Count do
169     Begin
```

```
170       chx:= chxList[i-1];
171
172       // Test si le checkbock est checked
173       if chx.Checked then
174         valeurs:= valeurs + IntToStr(i);
175     end;
176
177
178     Result:= valeurs;
179   end;
180
181   { ************************************************************************
182     *** Vérifie le format des dates                                      ***
183     *** @params String heure - Heure à vérifier                          ***
184     *** @Result boolean - Renvoi vrai si le format est bon               ***
185     ************************************************************************ }
186   function verifiFormatHeure(heure: string): boolean;
187   var
188     OutPutString: TStringList;
189     bon: boolean;
190   Begin
191     bon:= false;
192
193     OutPutString:= TStringList.Create;
194     OutPutString:= Split(heure, HEURE_SEPARATEUR);
195
196     if (StrToInt(OutPutString[0]) < MAX_HEURES) and
197       (StrToInt(OutPutString[1]) < MAX_MINUTES) then
198       bon:= true;
199
200     Result:= bon;
201   end;
202
203   { ************************************************************************
204     *** Change l'état des éléments                                       ***
205     *** @params Boolean active - true ou false                           ***
206     ************************************************************************ }
207   procedure TFrmGestionSeances.changeEtat(active: boolean);
208   Begin
209     LbxListeSeances.Enabled:= active;
210     BtnModifier.Enabled:= active;
211     BtnAjouter.Enabled:= active;
212     BtnSupprimer.Enabled:= active;
213   end;
214
215   { ************************************************************************
216     *** Charge et affiche les seances depuis un fichier ini              ***
217     ************************************************************************ }
218   procedure TFrmGestionSeances.chargeListeSeances();
219   var
220     fichierIni : TIniFile;
221     Sections, dataSection : TStringList;
222     ligneFormate, jourDiff, horaires: String;
223     i: integer;
224   Begin
225     //Initialisation
226     SetLength(listeSeances, 250);
```

```
227     LbxListeSeances.Clear;
228
229     // Lecture du fichier INI
230     fichierIni:= TIniFile.Create(FICHIER_SEANCES);
231     Sections:= TStringList.Create;
232     dataSection:= TStringList.Create;
233
234     fichierIni.ReadSections(Sections);
235
236     for i:= 0 to Sections.Count - 1 do
237     Begin
238       fichierIni.ReadSection(Sections[i], dataSection);
239       jourDiff:= '';
240       horaires:= '';
241
242       // Récupération des données du fichier ini
243       ListeSeances[i].section:= Sections[i];
244       ListeSeances[i].film:= fichierIni.ReadString(Sections[i], dataSection[0], 'N/A');
245       ListeSeances[i].salle:= fichierIni.ReadString(Sections[i], dataSection[1], 'N/A');
246       ListeSeances[i].jourDiff:= fichierIni.ReadString(Sections[i], dataSection[2],
          'N/A');
247       ListeSeances[i].heure1:= fichierIni.ReadString(Sections[i], dataSection[3], 'N/A'
          );
248       ListeSeances[i].heure2:= fichierIni.ReadString(Sections[i], dataSection[4], 'N/A'
          );
249       ListeSeances[i].heure3:= fichierIni.ReadString(Sections[i], dataSection[5], 'N/A'
          );
250       ListeSeances[i].heure4:= fichierIni.ReadString(Sections[i], dataSection[6], 'N/A'
          );
251       ListeSeances[i].diffuser:= fichierIni.ReadString(Sections[i], dataSection[7],
          'N/A');
252
253       // Modification des jours (Affichage en tout lettre)
254       jourDiff:= assembleJour(listeSeances[i].jourDiff, ',');
255
256       // Assemble les heures
257       if ListeSeances[i].heure1 <> '' then
258       Begin
259         if Length(horaires) > 0 then
260           horaires:= horaires + ',';
261
262         horaires:= horaires + listeSeances[i].heure1;
263       end;
264
265       if ListeSeances[i].heure2 <> '' then
266       Begin
267         if Length(horaires) > 0 then
268           horaires:= horaires + ',';
269
270         horaires:= horaires + listeSeances[i].heure2;
271       end;
272
273       if ListeSeances[i].heure3 <> '' then
274       Begin
275         if Length(horaires) > 0 then
276           horaires:= horaires + ',';
277
```

```
278         horaires:= horaires + listeSeances[i].heure3;
279       end;
280
281       if ListeSeances[i].heure4 <> '' then
282       Begin
283         if Length(horaires) > 0 then
284           horaires:= horaires + ',';
285
286         horaires:= horaires + listeSeances[i].heure4;
287       end;
288
289       // Formate le text pour l'affichage
290       ligneFormate:= AjusterText(ListeSeances[i].film, 15) + AjoutEspace(15);
291       ligneFormate:= ligneFormate + AjusterText(ListeSeances[i].salle, 10) +
          AjoutEspace(10);
292       ligneFormate:= ligneFormate + AjusterText(jourDiff, 51) + AjoutEspace(10);
293       ligneFormate:= ligneFormate + AjusterText(horaires, 23);
294
295
296       LbxListeSeances.Items.Add(ligneFormate);
297     end;
298
299     fichierIni.Free;
300   end;
301
302   { ************************************************************************
303     *** Ecrit les données d'un fichier dans un combobox                  ***
304     *** @params TCombobox cbs - Combobox qui reçois les données          ***
305     *** @params string fichier - Chemin du fichier                       ***
306     ************************************************************************ }
307   procedure donneDansComboBox(cbx: TComboBox; fichier: String);
308   var
309     f: TextFile;
310     ligne: string;
311     OutPutList : TStringList;
312     premiereLigne: Boolean;
313   Begin
314     premiereLigne:= true;
315     cbx.Clear;
316     OutPutList:= TStringList.Create;
317
318     // Test si le fichier existe
319     if FileExists(fichier) then
320     Begin
321       AssignFile(f, fichier);
322       reset(f);
323
324       repeat
325         readln(f, ligne);
326
327         if premiereLigne then
328           premiereLigne:= false
329         else
330         Begin
331           OutPutList:= Split(ligne, ';');
332           cbx.Items.Add(OutPutList[0]); // Met les données dans la combobox
333         end;
```

```
334
335       until eof(f);
336
337       OutPutList.free;
338       CloseFile(f);
339       cbx.ItemIndex:= 0;
340     end;
341   end;
342
343   { ************************************************************************
344     *** Sélectionne le bon élément                                       ***
345     *** @params TCombobox cbs - Combobox où l'on cherche notre élément   ***
346     *** @params string item - nom de l'élément choisi                    ***
347     ************************************************************************ }
348   procedure selectionneItemCombobox(cbx: TCombobox; item: string);
349   var
350     i: integer;
351   Begin
352     i:= 0;
353
354     while cbx.Items[i] <> item do
355     Begin
356       inc(i);
357     end;
358
359     cbx.ItemIndex:= i;
360   end;
361
362   { ************************************************************************
363     *** Ouvre les paramètres de modifications                            ***
364     ************************************************************************ }
365   procedure TFrmGestionSeances.BtnModifierClick(Sender: TObject);
366   var
367     index, i: integer;
368     tmpJour : integer;
369   begin
370
371     // Récupère l'index de l'élément sélectionner
372     index:= LbxListeSeances.ItemIndex;
373
374     // Charge les éléments des checkbox et sélectionne les bons éléments
375     donneDansComboBox(CbxFilm, FICHIER_FILMS);
376     selectionneItemCombobox(CbxFilm, listeSeances[index].film);
377     donneDansComboBox(CbxSalle, FICHIER_SALLES);
378     selectionneItemCombobox(CbxSalle, listeSeances[index].salle);
379
380     // Initialise les champs des horaires
381     edtHeure1.Text:= listeSeances[index].heure1;
382     edtHeure2.Text:= listeSeances[index].heure2;
383     edtHeure3.Text:= listeSeances[index].heure3;
384     edtHeure4.Text:= listeSeances[index].heure4;
385
386     // Initialise tous les checkbox a zéro
387     ChxLundi.Checked:= false;
388     ChxMardi.Checked:= false;
389     ChxMercredi.Checked:= false;
390     ChxJeudi.Checked:= false;
```

```
391     ChxVendredi.Checked:= false;
392     ChxSamedi.Checked:= false;
393     ChxDimanche.Checked:= false;
394
395     // Récupère les jours de diffusion
396     for i:= 1 to length(listeSeances[index].jourDiff) do
397     Begin
398       tmpJour:= StrToInt(listeSeances[index].jourDiff[i]);
399       case tmpJour of
400         1 : ChxLundi.Checked:= true;
401         2 : ChxMardi.Checked:= true;
402         3 : ChxMercredi.Checked:= true;
403         4 : ChxJeudi.Checked:= true;
404         5 : ChxVendredi.Checked:= true;
405         6 : ChxSamedi.Checked:= true;
406         7 : ChxDimanche.Checked:= true;
407       end;
408     end;
409
410
411     self.Height:= FRM_WIDTH_MAX;
412     changeEtat(false);
413   end;
414
415   { ************************************************************************
416     *** Annule les modification                                          ***
417     ************************************************************************ }
418   procedure TFrmGestionSeances.BtnAnuulerClick(Sender: TObject);
419   var
420     reponse : word;
421   begin
422     // Récupère la réponse
423     reponse:= MessageDlg('Etes-vous sûr de vouloir annuler les modifications ?',
      mtConfirmation, [mbYes, mbNo, mbCancel], 0);
424
425     // Test si la réponse est oui
426     if reponse = mrYes then
427     Begin
428       changeEtat(true);
429       self.Height:= FRM_WIDTH_MIN;
430     end;
431   end;
432
433   { ************************************************************************
434     *** Ouvre la fenêtre pour ajouter une séance                         ***
435     ************************************************************************ }
436   procedure TFrmGestionSeances.BtnAjouterClick(Sender: TObject);
437   var
438     CleVal: TCleVal;
439     NomSection: String;
440     chxList: TList;
441   begin
442     // Initialise les données de la fenêtre
443     donneDansComboBox(FrmAjouterSeance.CbxFilms, FICHIER_FILMS);
444     donneDansComboBox(FrmAjouterSeance.CbxSalles, FICHIER_SALLES);
445
446     with FrmAjouterSeance do
```

```
447     Begin
448       edtHeure1.Text:= '';
449       edtHeure2.Text:= '';
450       edtHeure3.Text:= '';
451       edtHeure4.Text:= '';
452
453       ChxLundi.Checked:= false;
454       ChxMardi.Checked:= false;
455       ChxMercredi.Checked:= false;
456       ChxJeudi.Checked:= false;
457       ChxVendredi.Checked:= false;
458       ChxSamedi.Checked:= false;
459       ChxDimanche.Checked:= false;
460
461     end;
462
463     if FrmAjouterSeance.showModal = mrOk then
464     Begin
465       // Initialisation des valeurs a sauvegarder dans le fichier INI
466       CleVal[0][0]:= 'Film';
467       CleVal[0][1]:= FrmAjouterSeance.CbxFilms.Items[FrmAjouterSeance.CbxFilms.
          ItemIndex];
468
469       CleVal[1][0]:= 'Salle';
470       CleVal[1][1]:= FrmAjouterSeance.CbxSalles.Items[FrmAjouterSeance.CbxSalles.
          ItemIndex];
471
472       CleVal[2][0]:= 'JourDiff';
473       CleVal[2][1]:= '';
474
475       chxList:= TList.Create;
476       chxList.Add(FrmAjouterSeance.ChxLundi);
477       chxList.Add(FrmAjouterSeance.ChxMardi);
478       chxList.Add(FrmAjouterSeance.ChxMercredi);
479       chxList.Add(FrmAjouterSeance.ChxJeudi);
480       chxList.Add(FrmAjouterSeance.ChxVendredi);
481       chxList.Add(FrmAjouterSeance.ChxSamedi);
482       chxList.Add(FrmAjouterSeance.ChxDimanche);
483       CleVal[2][1]:= jourEnNombre(chxList);
484       chxList.Free;
485
486       CleVal[3][0]:= 'Heure1';
487       CleVal[3][1]:= FrmAjouterSeance.edtHeure1.Text;
488
489       CleVal[4][0]:= 'Heure2';
490       CleVal[4][1]:= FrmAjouterSeance.edtHeure2.Text;
491
492       CleVal[5][0]:= 'Heure3';
493       CleVal[5][1]:= FrmAjouterSeance.edtHeure3.Text;
494
495       CleVal[6][0]:= 'Heure4';
496       CleVal[6][1]:= FrmAjouterSeance.edtHeure4.Text;
497
498       CleVal[7][0]:= 'Diffuser';
499       CleVal[7][1]:= '1';
500
501       //Compte le nombre de section existante
```

```
502       NomSection:= IntToStr(compteSectionIni(FICHIER_SEANCES) + 1);
503
504       if sauvegardeIni(FICHIER_SEANCES, NomSection, CleVal) then
505         MessageDlg('Séance anoutée avec succès !', mtInformation, [mbOk, mbCancel], 0)
506       else
507         MessageDlg('Une erreur est survenue lors de l''ajout !', mtError, [mbOk,
            mbCancel], 0);
508
509       chargeListeSeances();
510     end;
511   end;
512
513
514   { *************************************************************************
515     *** Procedure de la validation de la modification                      ***
516     ************************************************************************* }
517   procedure TFrmGestionSeances.BtnValiderModificationClick(Sender: TObject);
518   var
519     index: integer;
520     chxList: TList;
521     CleVal: TCleVal;
522   begin
523     // Initialisation
524     index:= LbxListeSeances.ItemIndex;
525
526     // Initialisation des valeurs a sauvegarder dans le fichier INI
527     CleVal[0][0]:= 'Film';
528     CleVal[0][1]:= CbxFilm.Items[CbxFilm.ItemIndex];
529
530     CleVal[1][0]:= 'Salle';
531     CleVal[1][1]:= CbxSalle.Items[CbxSalle.ItemIndex];
532
533     CleVal[2][0]:= 'JourDiff';
534     chxList:= TList.Create;
535     chxList.Add(ChxLundi);
536     chxList.Add(ChxMardi);
537     chxList.Add(ChxMercredi);
538     chxList.Add(ChxJeudi);
539     chxList.Add(ChxVendredi);
540     chxList.Add(ChxSamedi);
541     chxList.Add(ChxDimanche);
542     CleVal[2][1]:= jourEnNombre(chxList);
543     chxList.Free;
544
545     CleVal[3][0]:= 'Heure1';
546     CleVal[3][1]:= edtHeure1.Text;
547
548     CleVal[4][0]:= 'Heure2';
549     CleVal[4][1]:= edtHeure2.Text;
550
551     CleVal[5][0]:= 'Heure3';
552     CleVal[5][1]:= edtHeure3.Text;
553
554     CleVal[6][0]:= 'Heure4';
555     CleVal[6][1]:= edtHeure4.Text;
556
557     CleVal[7][0]:= 'Diffuser';
```

```
558     CleVal[7][1]:= '1';
559
560     // Sauvegarde dans le fichier
561     if sauvegardeIni(FICHIER_SEANCES, listeSeances[index].section, CleVal) then
562       MessageDlg('Modification effectuée avec succès !', mtInformation, [mbOk, mbCancel
          ], 0)
563     else
564       MessageDlg('Une erreur est survenue lors de la suppression !', mtError, [mbOk,
          mbCancel], 0);
565
566     chargeListeSeances();
567     LbxListeSeances.ItemIndex:= index;
568     changeEtat(true);
569     Self.Height:= FRM_WIDTH_MIN;
570
571   end;
572
573   { *************************************************************************
574     *** Procedure de la suppression                                       ***
575     ************************************************************************* }
576   procedure TFrmGestionSeances.BtnSupprimerClick(Sender: TObject);
577   var
578     index, i: integer;
579     CleVal: TCleVal;
580     f: TextFile;
581     reponse: word;
582   begin
583     // Demande confirmation
584     reponse:= MessageDlg('Voulez-vous vraiment supprimer cette séances ?', mtWarning, [
      mbYes,mbNo], 0);
585
586     // Test la réponse de la boîte de dialogue
587     if reponse = mrYes then
588     Begin
589       // Initialisation des variables
590       index:= LbxListeSeances.ItemIndex;
591
592       // "Suppresion" de la séance
593       ListeSeances[index].film:= '';
594       ListeSeances[index].salle:= '';
595       ListeSeances[index].jourDiff:= '';
596       listeSeances[index].heure1:= '';
597       listeSeances[index].heure2:= '';
598       listeSeances[index].heure3:= '';
599       listeSeances[index].heure4:= '';
600       listeSeances[index].diffuser:= '';
601
602       // Efface le fichier
603       if FileExists(FICHIER_SEANCES) then
604       Begin
605         AssignFile(f, FICHIER_SEANCES);
606         Rewrite(f);
607         CloseFile(f);
608       end;
609
610       index:= 1;
611
```

```
612       // Nettoyage du tableau (Enlève l'élément vide)
613       for i:= 0 to length(listeSeances) - 1 do
614       Begin
615         if listeSeances[i].film <> '' then
616         Begin
617
618           // Initialisation des valeurs a sauvegarder dans le fichier INI
619           CleVal[0][0]:= 'Film';
620           CleVal[0][1]:= listeSeances[i].film;
621
622           CleVal[1][0]:= 'Salle';
623           CleVal[1][1]:= listeSeances[i].salle;
624
625           CleVal[2][0]:= 'JourDiff';
626           CleVal[2][1]:= listeSeances[i].jourDiff;
627
628           CleVal[3][0]:= 'Heure1';
629           CleVal[3][1]:= listeSeances[i].heure1;
630
631           CleVal[4][0]:= 'Heure2';
632           CleVal[4][1]:= listeSeances[i].heure2;
633
634           CleVal[5][0]:= 'Heure3';
635           CleVal[5][1]:= listeSeances[i].heure3;
636
637           CleVal[6][0]:= 'Heure4';
638           CleVal[6][1]:= listeSeances[i].heure4;
639
640           CleVal[7][0]:= 'Diffuser';
641           CleVal[7][1]:= listeSeances[i].diffuser;
642
643           sauvegardeIni(FICHIER_SEANCES, IntToStr(index), CleVal);
644           inc(index);
645         end;
646       end;
647
648       MessageDlg('La séance a bien été supprimée !', mtInformation, [mbOk, mbCancel], 0
          );
649       chargeListeSeances();
650     end;
651   end;
652
653   { *************************************************************************
654     *** Test de la valeur des champs text                                 ***
655     ************************************************************************* }
656   procedure TFrmGestionSeances.edtHeure1Change(Sender: TObject);
657   begin
658     if ((length(edtHeure1.Text) > 0) or (length(edtHeure2.Text) > 0) or
659       (length(edtHeure3.Text) > 0) or (length(edtHeure4.Text) > 0)) and
660         (ChxLundi.Checked or ChxMardi.Checked or ChxMercredi.Checked or ChxJeudi.Checked
661           or ChxVendredi.Checked or ChxSamedi.Checked or ChxDimanche.Checked) then
662       BtnValiderModification.Enabled:= true
663     else
664       BtnValiderModification.Enabled:= false;
665   end;
666
667   { *************************************************************************
```

```
668     *** Filtre les caractères                                              ***
669     ************************************************************************** }
670   procedure TFrmGestionSeances.edtHeure1KeyPress(Sender: TObject;
671     var Key: Char);
672   begin
673     if not(key in['0'..'9', Chr(VK_BACK)]) then
674       key:= #0;
675   end;
676
677   { ************************************************************************
678     *** Procedure au relachement d'une touche                               ***
679     ************************************************************************** }
680   procedure TFrmGestionSeances.edtHeure1KeyUp(Sender: TObject; var Key: Word;
681     Shift: TShiftState);
682   begin
683     // Test si l'edit a 2 caractère et que la touche pressée n'est pas la touche VK_BACk
684     if (Length((Sender as TEdit).Text) = 2) and (not(key = ord(chr(VK_BACK)))) then
685     Begin
686       text:= (Sender as TEdit).Text;
687       (Sender as TEdit).Text:= text[1] + text[2] + HEURE_SEPARATEUR;
688     end;
689   end;
690
691   end.
692
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                ***
3    *** Fenêtre              : AjouterSeance                                ***
4    *** Auteur               : Devaud Alan                                  ***
5    *** Description          : Permet d'ajouter une séances                 ***
6    *** Version              : 1.0                                          ***
7    *** Date de création     : 05.05.2015                                   ***
8    *************************************************************************** }
9  unit U_OneWayTickets_AjouterSeance;
10
11 interface
12
13 uses
14   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
15   StdCtrls;
16
17 type
18   TFrmAjouterSeance = class(TForm)
19     Label1: TLabel;
20     Label2: TLabel;
21     CbxFilms: TComboBox;
22     CbxSalles: TComboBox;
23     GroupBox2: TGroupBox;
24     ChxLundi: TCheckBox;
25     ChxSamedi: TCheckBox;
26     ChxVendredi: TCheckBox;
27     ChxDimanche: TCheckBox;
28     ChxMardi: TCheckBox;
29     ChxMercredi: TCheckBox;
30     ChxJeudi: TCheckBox;
31     GroupBox3: TGroupBox;
32     Label4: TLabel;
33     Label5: TLabel;
34     Label6: TLabel;
35     Label7: TLabel;
36     edtHeure1: TEdit;
37     edtHeure2: TEdit;
38     edtHeure3: TEdit;
39     edtHeure4: TEdit;
40     BtnAnnuler: TButton;
41     BtnValider: TButton;
42     procedure edtHeure1KeyUp(Sender: TObject; var Key: Word;
43       Shift: TShiftState);
44     procedure edtHeure1KeyPress(Sender: TObject; var Key: Char);
45     procedure edtHeure1Change(Sender: TObject);
46     procedure BtnAnnulerClick(Sender: TObject);
47   private
48     { Déclarations privées }
49   public
50     { Déclarations publiques }
51   end;
52
53 var
54   FrmAjouterSeance: TFrmAjouterSeance;
55
56 implementation
57
```

```
58  {$R *.DFM}
59
60  { ************************************************************************
61    *** Procedure au relachement d'une touche                            ***
62    ************************************************************************ }
63  procedure TFrmAjouterSeance.edtHeure1KeyUp(Sender: TObject; var Key: Word;
64    Shift: TShiftState);
65  var
66    text: string;
67  begin
68    // Test si l'edit a 2 caractère et que la touche pressée n'est pas la touche VK_BACk
69    if (Length((Sender as TEdit).Text) = 2) and (not(key = ord(chr(VK_BACK)))) then
70    Begin
71      text:= (Sender as TEdit).Text;
72      (Sender as TEdit).Text:= text[1] + text[2] + ':';
73    end;
74  end;
75
76  { ************************************************************************
77    *** Filtre les caractères                                            ***
78    ************************************************************************ }
79  procedure TFrmAjouterSeance.edtHeure1KeyPress(Sender: TObject;
80    var Key: Char);
81  begin
82    if not(key in['0'..'9', Chr(VK_BACK)]) then
83      key:= #0;
84  end;
85
86  { ************************************************************************
87    *** Test de la valeur des champs text                                ***
88    ************************************************************************ }
89  procedure TFrmAjouterSeance.edtHeure1Change(Sender: TObject);
90  begin
91    if ((length(edtHeure1.Text) > 0) or (length(edtHeure2.Text) > 0) or
92      (length(edtHeure3.Text) > 0) or (length(edtHeure4.Text) > 0)) and
93        (ChxLundi.Checked or ChxMardi.Checked or ChxMercredi.Checked or ChxJeudi.Checked
94          or ChxVendredi.Checked or ChxSamedi.Checked or ChxDimanche.Checked) then
95      BtnValider.Enabled:= true
96    else
97      BtnValider.Enabled:= false;
98  end;
99
100 { ************************************************************************
101   *** Procedure d'annulation                                           ***
102   ************************************************************************ }
103 procedure TFrmAjouterSeance.BtnAnnulerClick(Sender: TObject);
104 var
105   reponse : word;
106 begin
107   reponse:= MessageDlg('Etes vous sûr de vouloir annuler l''ajout ?', mtConfirmation,
      [mbYes, mbNo, mbCancel], 0);
108
109   if reponse = mrYes then
110     ModalResult:= mrCancel;
111 end;
112
113 end.
```

```
1  { ***************************************************************************
2    *** Projet               : OneWayTickets                                  ***
3    *** Auteur               : Devaud Alan                                    ***
4    *** Fenêtre              : OneWayTickets_Impression                       ***
5    *** Description          : Permet d'informé que l'impression est en corus***
6    *** Version              : 1.0                                            ***
7    *** Date de création     : 12.05.2015                                     ***
8    *************************************************************************** }
9  unit U_OneWayTickets_Impression;
10
11 interface
12
13 uses
14   Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
15   StdCtrls, ExtCtrls;
16
17 type
18   TFrmImpressionEnCours = class(TForm)
19     Label1: TLabel;
20     Timer1: TTimer;
21     procedure Timer1Timer(Sender: TObject);
22   private
23     { Déclarations privées }
24   public
25     { Déclarations publiques }
26     maxSecond: integer;
27     second: integer;
28   end;
29
30 var
31   FrmImpressionEnCours: TFrmImpressionEnCours;
32
33 implementation
34
35 {$R *.DFM}
36
37 { ***************************************************************************
38   *** Procedure du timer                                                    ***
39   *************************************************************************** }
40 procedure TFrmImpressionEnCours.Timer1Timer(Sender: TObject);
41 begin
42   if second = maxSecond then
43   Begin
44     Timer1.Enabled:= false;
45     self.Visible:= false;
46   end;
47   inc(second);
48
49 end;
50
51 end.
52
```

```
{ ***************************************************************************
  *** Projet               : OneWayTickets                                ***
  *** Auteur               : Devaud Alan                                  ***
  *** Fenêtre              : OneWayTickets_ReservationAvance              ***
  *** Description          : Pas implémentée                              ***
  *** Version              : 1.0                                          ***
  *** Date de création     : 12.05.2015                                   ***
  *************************************************************************** }
unit U_OneWayTickets_ReservationAvance;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ComCtrls;

type
  TFrmReservationAvance = class(TForm)
    Label1: TLabel;
    CbxFilm: TComboBox;
    GroupBox2: TGroupBox;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    lblNomFilm: TLabel;
    lblDuree: TLabel;
    mmoSynopsis: TMemo;
    GroupBox1: TGroupBox;
    Label4: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    lblPrixEnfants: TLabel;
    lblPrixAdultes: TLabel;
    lblPrixEAA: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    GroupBox3: TGroupBox;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    lblSalle: TLabel;
    lblHoraire: TLabel;
    lblPlacesRestantes: TLabel;
    EdtNbBilletsEnfants: TEdit;
    UDNBilletsEnfants: TUpDown;
    EdtNbBilletsAdultes: TEdit;
    UDNBilletsAdultes: TUpDown;
    EdtNbBilletsEAA: TEdit;
    UDNBilletsEAA: TUpDown;
    LbxSeance: TListBox;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    BtnValider: TButton;
    BtnAnnuler: TButton;
  private
```

```
58        { Déclarations privées }
59      public
60        { Déclarations publiques }
61      end;
62
63    var
64      FrmReservationAvance: TFrmReservationAvance;
65
66    implementation
67
68    {$R *.DFM}
69
70    end.
71
```