Video bestanden afspelen met Free Pascal en Lazarus

Michaël Van Canneyt

December 17, 2012

Abstract

In een vorige bijdrage toonden we hoe filmpjes konden opgenomen worden met lazarus op Windows. In deze bijdrage tonen we hoe een willekeurig filmpje kan afgespeeld worden met een Lazarus programma op Windows en Linux.

1 Inleiding

Veel mensen kennen VideoLan als een prima open source media player: Deze speler kan een groot aantal formaten afspelen en doet dat bovendien op een hoop platformen: Windows, Linux en Mac OS. Deze speler is te downloaden van

```
http://www.videolan.org/
```

Minder mensen weten dat de functionaliteit van de videolan speler beschikbaar is voor programmeurs door middel van een bibliotheek. Deze bibliotheek wordt samen met de speler geinstalleerd: de speler zelf maaakt er gebruik van.

De bibliotheek (het zijn er eigenlijk 2) kan in andere programmas gebruikt worden om video bestanden te spelen: Het window waarin de video getoond wordt kan geplaatst worden binnen een window dat door het programma beheerd wordt. Op deze manier kan een eigen front-end (GUI) voor de VideoLan player gemaakt worden.

Alle header bestanden van de videolan bibliotheek (libvlc) zijn vertaald naar Pascal. Rond de headers werd een kleine OOP laag gemaakt. Deze OOP laag is vrijwel identiek aan het object model dat VideoLan zelf gebruikt: Met behulp van deze 3 componenten kunnen video bestanden afgespeeld worden zonder dat er een GUI aan te pas komt: de videos zullen getoond worden in een losstaand venster, of full-screen.

Van de belangrijkste component in de VLC classes werd een descendent gemaakt die in Lazarus gebruikt kan worden: het enige wat wordt toegevoegd is de mogelijkheid om het video venster binnen een LCL control te tonen. Een gelijkaardige component werd gemaakt voor fpGUI: het alternatief voor de LCL dat door Graeme Geldenhuys onderhouden wordt.

Alle code is ingecheckt in het versie beheer van Free Pascal en Lazarus: subversion. De code kan dus samen met de ontwikkelversie (trunk) gedownload worden, in afwachting van de eerste officiele release die deze componenten zal bevatten. Lezers die met een stabiele versie van lazarus/Free Pascal willen proberen, kunnen de code ook downloaden.

Lazarus gebruikert kunnen het lazvlc package installeren. Na installatie verschijen 2 componenten op de component palette: Een playlist manager en een media player component.

Hoewel alles specifiek voor Free Pascal en Lazarus ontwikkeld werd, kan de code met wat kleine wijzigingen ook gecompileerd worden in Delphi.

2 Architectuur

Er zijn 3 units nodig om de VLC libraries te gebruiken.

libvlc Deze unit bevat de low-level vertaling van de libvlc header files (vlc.h). de functies van deze unit moeten normaal niet gebruikt worden, maar er zitten ook wat constantes en enumerated types in.

vlc Deze unit bevat de OOP laag rond de low-level API. De unit bevat 5 classes, die hieronder beschreven worden.

lclvlc Deze unit bevat de LCL versie van de VLC player component: `TLCLVLCPlayer`. Als het lazvlc package geinstalleerd is in de IDE, wordt deze component getoond op de component palette.

De vlc unit bevat verschillende classes:

TVLCLibrary Deze class omkapselt de geladen VLC bibliotheek. Het is een dunne laag rond het VLC `libvlc_instance_t` type. De locatie en de globale opties van de bibliotheek kunnen hier ingesteld worden. Er zitten ook wat methodes en properties in die bv. de versie van de bibliotheek ophalen.

TVLCMediaPlayer Dit is de eigenlijke media player: De component die de videos toont. It contains the Play, Pause, Resume and Stop methods one expects from a video player component. The component is a wrapper around the VLC `libvlc_media_player_t` type.

TVLCMediaItem Deze class (een descendent van `TCollectionItem`) stemt overeen met 1 video of audio bestand of stream, die gespeeld kan worden door de `TVLCMediaPlayer` class. De belangrijkste property is het `Path`, de locatie van het bestand dat gespeeld moet worden. De class heeft ook een `URL` property en een `FileDescriptor` property voor remote media (een video stream). Deze collection item class is een laagje rond het `libvlc_media_t` type in Libvlc.

TVLCMediaItems is de collection die overeenstemt met de `TVLCMediaItem` items. Het is een OOP laagje rond het `libvlc_media_list_t` type.

TVLCMediaListPlayer is en afspeellijst component. De afspeellijst is een `TVLCMediaItems` collection, en een instance van `TVLCMediaPlayer` wordt gebruikt om de items in de lijst af te spelen. Deze class is een OOP laag rond het VLC `libvlc_media_list_player_t` type.

Uit deze beschrijving zou het duidelijk moeten zijn dat voor elk type in de VLC library er een corresponderende pascal klasse is.

3 De bibliotheek laden en initialiseren

De VLC bibliotheek wordt dynamisch (on demand) geladen. Na het laden moet ze geinitialiseerd worden. Dit proces wordt door de `TVLCLibrary` component gestuurd.

Onder normale omstandigheden hoeft er geen instance van `TVLCLibrary` gecreëerd worden. De `VLCLibrary` functie geeft een enkele (globale) instance van deze klasse terug.

Deze instance zal de VLC library laden zodra de `Initialize` methode opgeroepen wordt. Deze methode wordt automatisch opgeroepen door de andere classes wanneer het nodig is. Ze hoeft doorgaans dus niet expliciet opgeroepen worden.

De `LibraryPath` property kan ingesteld worden op de naam en het pad naar de VLC library. Indien gezet, neemt het systeem aan dat de core library (de 2de library die VLC gebruikt) op dezelfde locatie staat. Indien ze niet gezet is, zal het systeem een standaard naam en locatie gebruiken. De `LibraryArgs` stringlist kan gebruik worden om een reeks argumenten aan de library door te geven. Deze argumenten zijn dezelfde argumenten die op de commandolijn van de VLC player gebruikt kunnen worden.

Bij het initialiseren van de bibliotheek moet aan 2 punten aandacht geschonken worden:

1. De library laadt enkele video drivers. Sommige van deze drivers veroorzaken floating point errors, die naar exceptions worden omgezet. Deze exceptions kunnen uitgeschakeld worden met de `SetExceptionMask` functie in de Math unit, als volgt:

   ```
   setexceptionmask([exInvalidOp, exDenormalized, exZeroDivide,
                     exOverflow, exUnderflow, exPrecision]);
   ```

   De exceptions kunnen terug worden ingeschakeld met:

   ```
   setexceptionmask([]);
   ```

   Dit niet doen zal alle exceptions, veroorzaakt door fouten in floating point berekeningen uitschakelen.

2. De VLC library gebruikt threads. Aangezien deze threads buiten de Free Pascal RTL aangemaakt worden, weet de RTL niet van het bestaan van deze threads, en wordt het threading niet geinitialiseerd. Dit veroorzaakt fouten indien de VLC callbacks (event handlers) gebruikt worden, want deze callbacks worden vanuit een VLC thread opgeroepen. Als het programma dat de VLC library gebruikt zelf geen threads aanmaakt, kan het threading systeem toch als volgt correct geinitialiseerd worden:

   ```
   With TThread.Create(False) do
     Terminate;
   ```

Op windows maakt de VideoLan installatie een registry sleutel met daarin de locatie van de VLC installatie. Dit kan gebruikt worden om de correcte locatie van de bibliotheek in te stellen als volgt:

```
Procedure SetVLCLibraryPath;

Const
  VLCKey = 'Software\VideoLAN\VLC';
var
  D : String;

begin
  Result := '';
  With TRegistry.Create(KEY_READ) do
    try
      RootKey:=HKEY_LOCAL_MACHINE;
      if OpenKey(VLCKey,False) then
        if ValueExists('InstallDir') then
          begin
          D:=r.ReadString('InstallDir'));
          D:=IncludeTrailingPathDelimiter(D);
          VLCLibrary.LibraryPath:=D+libvlc.libname;
```

```
        end;
    finally
      Free;
    end;
  end;
end;
```

4 Een eenvoudige commando-lijn video player

Nadat de VLC library geinitialiseerd is, kan de `TVLCMediaPlayer` component gebruikt worden om een video of audio bestand af te spelen.

Deze class heeft de volgende methodes:

```
procedure SetMedia(M: TVLCMediaItem);
procedure Play;
Procedure Play(M : TVLCMediaItem);
Procedure PlayFile(Const AFileName : String);
Procedure Stop;
procedure Pause;
procedure Resume;
procedure NextFrame;
function Snapshot(Const AFileName: String): Boolean;
function Snapshot(Const AFileName: String;
                  AWidth, AHeight: Cardinal): Boolean;
function GetVideoSize(Var AWidth, AHeight: Cardinal): Boolean;
```

De betekenis ervan is intuitief duidelijk:

SetMedia Bereidt een media item voor om te spelen.

Play Speelt het media item af dat voorbereid werd met `SetMedia`. Als een media item doorgegeven wordt aan deze functie, wordt het voorbereid met SetMedia, en daarna afgespeeld.

PlayFile Zal een bestand afspelen. Het maakt een `TVLCMediaItem`, initialiseert het met de opgegeven bestandsnaam, en roept dan `Play` op.

Stop stopt de speler.

Pause Pauseert het afspelen.

Resume Start het afspelen op de plek waar het gepauseerd was.

NextFrame Springt naar het volgende frame in de video.

GetVideoSize Geeft de breedte en hoogte van de video die wordt afgespeeld.

Snapshot Maakt een beeldje (snapshot) van het huidige video frame en bewaart het in het opgegeven bestand. Als geen breedte en hoogte worden opgegeven worden ze berekend door `GetVideoSize`.

De controle commandos zoals play, resume werken asynchroon: deze commandos keren onmiddellijk terug. De reden hiervoor is dat VLC het werk in de achtergrond uitvoert. De huidige toestand van de VLC player kan opgevraagd worden door de volgende properties:

Playing Een boolean die aangeeft of de speler op dit ogenblik media aan het afspelen is.

State Een meer gedetailleerde status indicatie.

De status kan een van de volgende waardes zijn:

libvlc_NothingSpecial De speler doet niets.

libvlc_Opening De speler is een media bestand of stream aan het openen.

libvlc_Buffering De speler is data aan het bufferen.

libvlc_Playing De speler is een audio/video aan het afspelen

libvlc_Paused De speler is gepauseerd met `Pause`.

libvlc_Stopped De speler is gestopt met `Stop`.

libvlc_Ended De speler heeft het einde van het media bestand bereikt.

libvlc_Error Er is een fout opgetreden die niet opgevangen kon worden.

Met deze properties en methodes kan een minimalistisch commandolijn programma gemaakt worden dat een bestand afspeelt:

```
program testvlc;

{$mode objfpc}{$H+}

uses
  {$ifdef unix}cthreads,{$endif}
  sysutils, math, libvlc, vlc;

Const
  AllExceptions =
   [exInvalidOp, exDenormalized, exZeroDivide,
    exOverflow, exUnderflow, exPrecision];

begin
  SetExceptionMask(AllExceptions);
  With TVLCMediaPlayer.Create(Nil) do
    try
      PlayFile(ParamStr(1));
      Repeat
        Sleep(100);
      until State in [libvlc_Ended,libvlc_Error];
    finally
      Free;
    end;
end.
```

Het programma is extreem eenvoudig: een `TVLCMediaPlayer` player instance wordt gemaakt, en het eerste commandolijn argument wordt aan `PlayFile` doorgegeven. Dan start een lus die wacht tot de speler klaar is met spelen. Daarna wordt de player instance vrijgegeven en het programma stopt. Eenvoudiger dan dit is moeilijk te bereiken.

5 Een afspeellijst beheren

De `TVLCPlayer` component speelt 1 media item en stopt dan. De `TVLCMediaItem` instance kan deel uitmaken van een collectie van media items. De VLC library heeft de nodige functionaliteit om een afspeellijst te beheren en af te spelen. to maintain and play a collection in a playlist. Deze functionaliteit zit vervat in de `TVLCMediaListPlayer` component. Deze component heeft dezelfde commandos (Play, Pause, Stop) als de `TVLCMediaPlayer` component, maar heeft enkele bijkomende navigatie methodes zoals `Prev` en `Next` om naar de volgende of vorige items in de afspeellijst te springen.

De volgende properties zijn ook beschikbaar:

Player Dit dient ingevuld te worden met een instance van de `TVLCMediaPlayer` component. Deze component wordt dan gebruikt om de media items af te spelen.

PlayMode Deze (enumerated) property bepaalt hoe de items worden gespeeld. Mogelijke waarden zijn `pmNormal`, `pmLoop`, `pmRepeat`: De betekenis is duidelijk: normaal afspelen, in een lus of herhalen.

MediaItems Dit is de collection met `TVLCMediaItems` die gespeeld moeten worden.

Met deze component kan het commandolijn programma uitgebreid worden om een lijst bestanden te spelen in plaats van een enkel bestand:

```
program testvlc2;

{$mode objfpc}{$H+}

uses
  {$ifdef unix}cthreads,{$endif}
  sysutils, math, libvlc, vlc;

Const
  AllExceptions =
   [exInvalidOp, exDenormalized, exZeroDivide,
    exOverflow, exUnderflow, exPrecision];

Var
  i : integer;

begin
  SetExceptionMask(AllExceptions);
  with TVLCMediaListPlayer.Create(Nil) do
    try
      Player:=TVLCMediaPlayer.Create(Nil);
      For I:=1 to ParamCount do
        TVLCMediaItem(MediaItems.Add).Path:=ParamStr(i);
      Play;
      Repeat
        Sleep(100);
      until State in [libvlc_Ended,libvlc_Error];
    finally
      Player.Free;
      Free;
    end;
end.
```

Het enige dat moet worden ingesteld worden bij een `TVLCMediaItem` instance is de `Path` property.

6 Bijkomende properties en events gebruiken in een GUI

De commandos en voorbeelden die tot nu toe gegeven werden, tonen niet hoe de interactie met de speler gebeurt. Een bestandsnaam wordt doorgegeven, en het programma wacht tot de speler klaar is met afspelen.

Dit is uiteraard niet erg interessant, en in een echte applicatie wil de gebruiker wat meer controle over de speler.

Er zijn verschillende properties die gebruikt kunnen worden om de video of audio die wordt afgespeeld te onderzoeken - of zelfs te beïnvloeden:

AudioTrackCount Het aantal audio sporen in het video bestand. Alleen lezen.

AudioTrackDescriptions Een array van strings (0-gebaseed) die de audio sporen beschrijven. Alleen lezen.

AudioTrack Het audio spoor (0-gebaseerd) dat gebruikt moet worden wanneer de video getoond worden (lezen/schrijven).

AudioDelay Een vertraging (in milliseconden) die in acht genomen moet worden bij het spelen van audio (lezen/schrijven).

AudioVolume het audio volume. Een geheel getal tussen 0 en 200 (lezen/schrijven).

AudioMuted Een boolean die kan gezet worden om het geluid af te zetten (mute - lezen/schrijven).

Channel Het audio kanaal dat gebtuikt wordt.

De volgende properties geven wat informatie over de video:

ChapterCount het aantal hoofdstukken in de video. (lezen/schrijven).

Chapter Het huidige hoofdstuk. (lezen/schrijven).

VideoWidth De breedte van de video, in pixels. (alleen lezen).

VideoHeight De hoogte van de video, in pixels. (alleen lezen).

VideoLength De duur van de video, in milliseconden. (alleen lezen).

VideoDuration De duur van de video als een date/time value. (alleen lezen).

VideoPosition De huidige positie in het video bestand, in milliseconden. (lezen/schrijven).

VideoFractionalPosition De huidige positie in het video bestand, percentueel uitgedrukt. (lezen/schrijven).

VideoFramesPerSecond Het aantal frames per seconde. (alleen lezen).

VideoScale De schaal waarop de video getoond wordt, waar 1 normaal is.(lezen/schrijven).

AspectRatio Een string die de aspect ratio beschrijft (verhouding hoogte/breedte). Read/write.

Tot slot kunnen de volgende properties gezet worden om te bepalen hoe de video getoond wordt:

FullScreenMode Indien true, wordt de video full-screen getoond.

FitWindow Indien true, wordt het parent window van het video window herschaald zodat het de grootte van de video heeft.

UseEvents Indien true, worden de callbacks (event handlers) die libVLC voorziet ingesteld, en kunnen de event handlers gebruikt worden.

Er zijn een heleboel events beschikbaar:

OnMediaChanged Wordt opgerepen als het media bestand wijzigt voor de player. Dit is maar nuttig in combinatie met de playlist.

OnNothingSpecial Opgeroepen als de speler niets doet (idle).

OnBackward Wordt opgeroepen als de speler terugspoelt.

OnBuffering Wordt opgeroepen als de speler data buffert om later te spelen.

OnEOF Wordt opgeroepen als de speler het einde van de media bereikt.

OnError Wordt opgeroepen als libvlc een fout tegenkomt.

OnForward Wordt opgeroepen als de speler vooruitspoelt.

OnOpening Wordt opgeroepen als de speler een bestand opent.

OnPause Wordt opgeroepen als de speler gepauseerd wordt.

OnPlaying Wordt opgeroepen als de speler terug begint te spelen na pause.

OnStop Wordt opgeroepen als de speler gestopt wordt.

OnLengthChanged Wordt opgeroepen als de lengte van het video bestand wijzigt. De nieuwe lengte wordt doorgegeven aan de event handler. Dit wordt meestal maar 1 maal opgeroepen, zodra de lengte van de video vastligt.

OnTimeChanged Wordt opgeroepen als de positie (in tijd) in het video bestand wijzigt.

OnPausableChanged Wordt opgeroepen als de mogelijkheid tot pauzeren wijzigt. Normaal wordt dit slechts 1 maal opgeroepen per bestand.

OnPositionChanged Wordt opgeroepen als de positie (in percent) in het video bestand wijzigt.

OnSeekableChanged Wordt opgeroepen als de mogelijkheid tot positioneren wijzigt. Normaal wordt dit slechts 1 maal opgeroepen per bestand.

OnTitleChanged Wordt opgeroepen als de titel in het video bestand wijzigt.

OnSnapshot Wordt opgeroepen als er een beeldje (snapshot) gemaakt werd.

Deze events worden opgeroepen vanuit de videolan thread die instaat voor het afspelen van de video. Dat betekent dat als de events gebruikt worden om het scherm te updaten, bv. een trackbar op de correcte positie te zetten, of om de huidige en totale tijd van de video te tonen, dan moet dit gebeuren in een door `Synchronize` opgeroepen procedure.

De LCL versie van de control heeft een eigenschap `ParentWindow`. Deze kan ingesteld worden op elke `TWinControl` afgeleide die child controls aanvaardt, met als gevolg dat de video dan binnen deze control getoond zal worden. Meestal zal dit een `TPanel` zijn, of een `TForm`

Om enkele van deze properties te demonstreren, maken we een kleine video speler, bestaande uit een menu, toolbar, een panel voor het tonen van de video, en een panel met 2 trackbars: 1 voor volume, 1 voor het zetten van de positie in de video.

De toolbar en het menu bevatten wat knoppen of menu items om een bestand te openen, en om het afspelen te controleren; Afspelen, pauzeren en voortspelen. Alle knoppen en menu items hebben een `TAction` die de eigenlijke actie uitvoert.

In de `OnCreate` event van de form wordt de speler component gemaakt, en worden alle event handlers ingesteld:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  FPlayer:=TLCLVLCPlayer.Create(Self);
  FPlayer.ParentWindow:=PVideo;
  FPlayer.OnTimeChanged:=@DoTimeChanged;
  FPlayer.OnPositionChanged:=@DoPositionChanged;
  FPlayer.OnLengthChanged:=@DoLengthChanged;
  FPlayer.UseEvents:=True;
end;
```

Het panel waarin de video getoond wordt, noemt `PVideo`. Er zijn 3 events die gebruikt worden: 2 events om de tijd in en lengte (in minuten/seconden) van de video te tonen, en 1 om de positie van een trackbar up to date te brengen.

Het spelen van een video bestand (de `AOpen` actie) is heel eenvoudig:

```
 procedure TMainForm.MIOpenClick(Sender: TObject);
begin
  With ODVideo do
   begin
   FileName:=FFileName;
   if Execute then
     begin
     FFileName:=FileName;
     FPlayer.PlayFile(FFileName);
     Caption:='Lazarus video demo: '+FFIleName;
     end;
   end;
end;
```

Om de display te updaten terwijl de video aan het afspelen is, zijn er 3 event handlers nodig: Wanneer VLC een wijziging in de positie rapporteert door `OnPositionChanged`, wordt de `TBVideo` trackbar positie bijgewerkt als volgt:

```
procedure TMainForm.DoPositionChanged(Sender: TObject; const APos: Double);
begin
  FNewPosition:=Round(APos*100);
  TThread.Synchronize(Nil,@SetNewPosition);
end;
```

De nieuwe positie wordt bewaard: de fractionele waarde wordt omgezet in een waarde tussen 1 en 100. De event handler wordt opgeroepen in een thread, dus moet `Synchronize` gebruikt worden om `SetNewPosition` op te roepen:

```
procedure TMainForm.SetNewPosition;
```

```
begin
  FShowing:=True;
  try
    TBVideo.Position:=FNewPosition;
  finally
    FShowing:=False;
  end;
end;
```

De variable `FShowing` wordt gebruikt om in de `OnChange` event handler van de trackbar na te gaan of de positie door de video wordt gezet. op deze wijze kan de event handler beslissen of het de gebruiker is die de positie wijzigt, of dat het de video speler component is die zijn nieuwe positie meedeelt:

```
procedure TMainForm.TBVideoChange(Sender: TObject);
begin
  if not FShowing then
    FPlayer.VideoFractionalPosition:=TBVideo.Position/100;
end;
```

Merk op dat de positie als fractionele waarde gezet moet worden (met 0=start en 1=einde van de video).

OP gelijkaardige wijze worden de totale en huidige tijd afgebeeld met de `OnTimeChanged` en `OnLengthChanged` event handlers:

```
procedure TMainForm.DoTimeChanged(Sender: TObject;
                                  const time: TDateTime);
begin
  FCurrentTime:=Time;
  TThread.Synchronize(Nil,@DisplayTime);
end;

procedure TMainForm.DoLengthChanged(Sender: TObject;
                                    const time: TDateTime);
begin
  FNewLength:=Time;
  TThread.Synchronize(Nil,@DisplayTime);
end;
```

Beide event handlers passen de caption van een label aan in de `DisplayTime` routine:

```
procedure TMainForm.DisplayTime;

  Function TtoS (T : TDateTime) : string;
  Var
    h,m,s,ms : Word;
  begin
    DecodeTime(T,h,m,s,ms);
    if h>0 then
      Result:=FormatDateTime('hh:nn:ss',T)
    else
      Result:=FormatDateTime('nn:ss',T);
  end;
```

```
Var
  s : string;

begin
  S:='/';
  if FNewLength>0 then
    S:=S+TToS(FNewLength)
  else
    S:=S+'?';
  if (FCurrentTime>0) then
    S:=TToS(FCurrentTime)+S
  else
    S:='0:0'+S;
  LTime.Caption:=S;
end;
```

Deze routine doet niets anders dan een leesbare time/length tonen in een label.

De `OnExecute` handler van de Stop, Pause en Resume acties zijn rechttoe-rechtaan:

```
procedure TMainForm.BStopClick(Sender: TObject);
begin
  FPlayer.Stop;
end;

procedure TMainForm.BPauseClick(Sender: TObject);
begin
  FPLayer.Pause;
end;

procedure TMainForm.BResumeClick(Sender: TObject);
begin
  FPlayer.Resume;
end;
```

De `OnUpdate` event handlers van deze acties maken gebruik van de `State` property van de video player component:

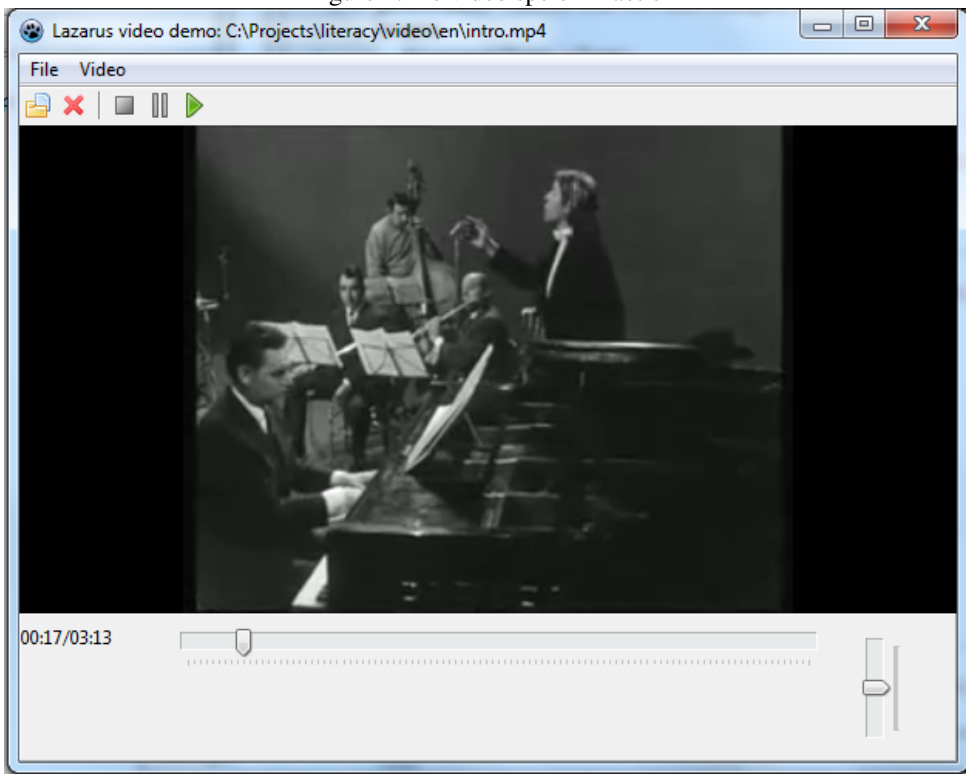
```
procedure TMainForm.AStopUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled:=FPlayer.Playing;
end;

procedure TMainForm.APauseUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled:=FPlayer.Playing;
end;

procedure TMainForm.AResumeUpdate(Sender: TObject);
begin
  (Sender as TAction).Enabled:=FPlayer.State=libvlc_Paused;
end;
```

Het enige wat nog gedaan moet worden is de volumeregeling door middel van een trackbar (`TBVolume`):

Figure 1: De video speler in actie




```
procedure TMainForm.TBVolumeChange(Sender: TObject);
begin
  FPlayer.AudioVolume:=TBVolume.Position;
end;
```

Met dit alles is een complete video speler geprogrammeerd. Het resultaat is zichtbaar in figure 1 on page 12.

7 Conclusion

De videolan bibliotheek is een goede keuze voor het inwerken van een video speler in een cross-platform applicatie: Videolan ondersteunt een groot aantal video formaten, kan streams spelen van andere computers in het netwerk, en heeft een eenvoudige API, die dan nog eens op vrij natuurlijke wijze in Object Pascal klassen vertaald is. Doordat het mee verdeeld wordt met Free Pascal en Lazarus is het waarschijnlijk de eenvoudigste manier om video te tonen in een cross-platform Lazarus programma.