The Dragon FlicPanel for Win95 and NT
(ActiveX control and Delphi component)

v1.55c

(c) Dr P W Kuczora

27th May 1998

Overview:

The Dragon FlicPanel is a 32-bit component for Borland Delphi 2 and 3 which replicates the functionality of the 16-bit AAPLAY.DLL flic-playing DLL which was provided by Autodesk Inc. It is also available as an ActiveX Control for use with Visual BASIC, VC++ and Internet Explorer, along with all other applications which support the ActiveX format. The component is written in pure 32-bit Delphi, and uses a combination of Device Independent Bitmaps (DIBs) and multi-threading for optimal performance. As far as possible, the functionality of AAPLAY.DLL and the 16-bit Dragon FlicPlayer component has been replicated in the new component.

Versions:

The ActiveX control is supplied in the following versions:
1. Unregistered shareware ocx — shows an “about” screen whenever a flic file is opened
2. Registered version of ocx — disables “about” screen

The Delphi component is supplied in the following versions:
1. Unregistered shareware dcu — shows an “about” screen whenever a flic file is opened
2. Full source code

Availability:

Via WWW:

The primary Web site for the ActiveX control is:

Windows95.com -
http://www.windows95.com

The primary Web site for the Delphi component is:

The Delphi Super Page -
http://sunsite.icm.edu.pl/delphi/
http://www.cdrom.com/pub/delphi_www/

There is also a small **home page** for the component at:
http://pobox.com/~dragon.enterprises/flicpanel/

Registration and payment:

PLEASE NOTE:

The Dragon FlicPanel is distributed electronically only, via MIME-encoded email.

Registration via WWW:

Registration and payment will be available from the component home page at:
http://pobox.com/~dragon.enterprises/flicpanel/

This URL is redirected via a forwarding service, so it will remain current whatever ISP I am using.

Registration by mail:

FlicPanel Registration
1 Cliffside
69 St Mildreds Road
Westgate on Sea
Kent
CT8 8RL England

Please make cheques / money orders payable to "B Woodward".

Be sure to include details of the email address to which the component should be sent.

Pricing details:

	Cheque drawn on UK bank International Money Order Eurocheque	Internet Visa
Delphi full source code (Delphi 2 and 3)	60 UKP or $90	$99
Registered ActiveX version	30 UKP or $45	$55

Installation:

Delphi 2:

The **file *flcpanl2.zip*** contains the Delphi2 component version in the following files:
flicpanelreg.pas *flicpanelreg.dcr* *flicpanel.dcu* *demo2.zip* *flcpanel.doc*

Place *flicpanelreg.pas*, *flicpanelreg.dcr* and *flicpanel.dcu* into a suitable directory and install the component as per usual. By default, the component installs to Samples section of the component palette. Edit *flicpanelreg.pas* to change this to suit your setup.

Delphi 3:

The **file *flcpanl3.zip*** contains the Delphi3 package version in the following files:
flicpanelreg.pas *flicpanelreg.dcr* *flicpanel.dcu* *demo3.zip* *flcpanel.doc*

Place *flicpanelreg.pas*, *flicpanelreg.dcr* and *flicpanel.dcu* into a suitable directory and install the component into a new or existing package as required. By default, the component installs to Samples section of the component palette. Edit *flicpanelreg.pas* to change this to suit your setup.

Demo:

Place the demo zip file in its own directory and unzip it to produce *flicpaneltest.dpr* and its associated files. The demo is fairly self-explanatory, if a little rough around the edges. The notify check-boxes can be used to trigger wav files at a given frame each time round the playback loop - play/pause or step to the required frame, click a check-box and select a wav file.

ActiveX:

The **file *flicpanelx.zip*** contains the ActiveX control in the following files:
flicpanelxcontrol.ocx *flcpanel.doc*

Install this as per the instructions supplied with the application that you wish to use.

Documentation:

The **file *flicpaneldoc.zip*** contains a generic set of documentation in the following file:
flcpanel.doc

Legal stuff :

RIGHTS: Title, ownership rights, and intellectual property rights in and to the Software shall remain in P W Kuczora and/or his suppliers. The Software is protected by the copyright laws of the United Kingdom and international copyright treaties. This License gives you no rights to such content.

DISCLAIMER: This code is for demonstration purposes only and should be used at your own risk. The Software is provided on an "AS IS" basis, without warranty of any kind, including without limitation the warranties of merchantability, fitness for a particular purpose and non-infringement. The entire risk as to the quality and performance of the Software is borne by you. Should the Software prove defective, you and not P W Kuczora assume the entire cost of any service and repair. In addition, you must determine that the Software sufficiently meets your requirements. This disclaimer of warranty constitutes an essential part of the agreement.

SOME U.S. STATES DO NOT ALLOW EXCLUSIONS OF AN IMPLIED WARRANTY, SO THIS DISCLAIMER MAY NOT APPLY TO YOU AND YOU MAY HAVE OTHER LEGAL RIGHTS THAT VARY FROM STATE TO STATE OR BY JURISDICTION.

LIMITATIONS: LIABILITY -- UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, TORT, CONTRACT, OR OTHERWISE, SHALL P W Kuczora OR HIS SUPPLIERS OR RESELLERS BE LIABLE TO YOU OR ANY OTHER PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES. IN NO EVENT WILL P W Kuczora BE LIABLE FOR ANY DAMAGES IN EXCESS OF P W Kuczora's LIST PRICE FOR A LICENSE TO THE SOFTWARE, EVEN IF P W Kuczora SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. FURTHERMORE, SOME U.S. STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION AND EXCLUSION MAY NOT APPLY TO YOU.

Using the component:

The FlicPanel is intended to be driven from software in multimedia applications, using function calls which correspond as closely as possible to those used by the earlier 16-bit AAPLAY.DLL.

The essential steps in using the component are:

1. Set the Filename property.
2. Open the panel.
3. Set the CallbackHandle property.
4. Issue your flic playing commands.
5. Close the panel.

Constants and data types:

The following constants are defined:

const:
NULL = 0
NULLPTR = ^0
AA_STOPPED = 1
AA_QUEUED = 2
AA_PLAYING = 3
AA_PAUSED = 4
AA_DONE = 5

The following data types are declared:

type:

AASPEED = word;	Type that holds speed for animation
dword = longint;	No 32-bit unsigned in Object Pascal
SoundString = string;	

History:

27th May 1998	v1.55	Notifications arrays dynamic resize now sends status message
27th May 1998	v1.54	Thread priority property added
26th May 1998	v1.53	Notifications arrays now dynamically resize as required
26th May 1998	v1.52	Notifications can be set to repeat every loop (if Loop = 0)
26th May 1998	v1.51	Added callback message every frame when StatusMessages = TRUE Modified demo program to use new functionality
26th May 1998	v1.50	Core routines re-implemented within thread code Key variables moved into FlicVars data structure COMPONENT IS NOW THREAD-SAFE
24th May 1998	v1.28	Key parameters moved into FlicParams data structure
22nd May 1998	v1.27	Version property added
21st May 1998	v1.26	Pascal files cause access violations opening read-only files! Now use API routines FileOpen, FileRead, FileClose
19th May 1998	v1.25	Stop now uses WaitForSingleObject and GetExitCodeProcess Thread is killed if doesn't terminate itself
18th May 1998	v1.24	Changed registration method
17th May 1998	v1.23	Play thread now yields time-slice between frames
17th May 1998	v1.22	Fixed bug where thread failed to terminate correctly
21st Jan 1998	v1.21	Added conditional compilation for registered version
2nd Nov 1997	v1.20	Further optimisation of inner rendering loops
1st Nov 1997	v1.18	flicsurf.pas subsumed into flicpanel.pas as an include file
31st Oct 1997	v1.17	AutoPlay property added.
6th Oct 1997	v1.16	Conditional compilation for D2 and D3 versions added.
5th Oct 1997	v1.15	Now decodes COLOR (type 11) palettes from .fli files as well as COLOR256 (type 4) from .flc
1st Oct 1997	v1.14	Now derived from TCustomControl, not TGraphicControl First ActiveX version generated from this version
30th Sept 1997	v1.13	Added Stretchable property - uses StretchBlt to render DebugMessages property removed
29th Sept 1997	v1.12	Now derived from TGraphicControl, not TControl
29th Sept 1997	v1.11	D3 NULL bugs in FlicSurface sorted out
19th Sept 1997	v1.10	first pass at optimising rendering algorithms
12th Sept 1997	v1.00	initial development

Procedures and Properties:

published

```
property NoPaintLoad: boolean;
property HideFlicWindow: boolean;
property NotifyMessages: boolean;
property StatusMessages: boolean;
property Stretchable: boolean;
property AutoPlay: boolean;
property Version: string; (read-only)
property ThreadPriority: TThreadPriority;
```

public

```
constructor Create(Owner: TComponent); override;
destructor Destroy; override;
procedure Open;
procedure Reload(mode, mask: word);
procedure Play;
procedure Close;
procedure Stop;
procedure Pause;
procedure Resume;
procedure Step;
procedure Back;
procedure Previous;
procedure Next;
procedure Eject;
procedure Notify(Loop, Frame, MsgNum: integer);
procedure Cancel(TopLoop, TopFrame, BottomLoop, BottomFrame: integer);
procedure CancelAll;
procedure Paint; virtual;

property  Canvas: TCanvas;
property  FlicSurface : TFlicSurface;
property NotifyMessageNum: word;
property StopMessageNum: word;
property FrameMessageNum: word;
property RedimMessageNum: word;
property FlicHeight: integer;
property FlicWidth: integer;
property DesignSpeed: AASPEED;
property Speed: AASPEED;
property CallbackHandle: THandle;
property X: integer;
property Y: integer;
property CX: integer;
property CY: integer;
property OrgX: integer;
property OrgY: integer;
property CurrentFrame: integer;
property CurrentLoop: integer;
property PlayLoops: dword;
property PlayFrames: dword;
property PauseTime: word;
property Sound: SoundString;
property SoundDelay: integer;
property IsOpen: boolean;
property FlicLoaded: boolean;
property LastError: integer;
property Status: word;
```

procedure Open;

This procedure opens the panel and loads the animation specified in the Filename property with the published properties as specified.

Note that published properties should be assigned before the panel is opened, while public properties should be assigned once an animation has been opened.

procedure Reload(mode, mask: word);

Loads another animation over an existing one.

First, set the Filename property to that of the animation file to be opened, then call Reload.

The existing animation must be stopped, or have finished playing in order to make this call.

Parameters:

mode:

Ignored in this version of the panel

mask:

Ignored in this version of the panel

procedure Play;

Plays the animation loaded by Open or ReLoad. Animation is carried out by a separate program thread.

Play begins from the current position (CurrentLoop and CurrentFrame).

Play returns after the animation has begun playing.

procedure Close;

This simply calls Eject.

procedure Stop;

Stops the playing animation began by Play, by destroying the animation thread.

Stopped animations begin with the next frame of the animation, when they are replayed.

You may also set CurrentLoop and/or CurrentFrame to start a stopped animation at a frame other than the first frame.

procedure Pause;

Pauses a playing animation by suspending the animation thread.

A paused animation is still considered playing, so no other full screen animations will play.

Paused animations will continue playing from the current frame, when they are restarted by calling Resume.

procedure Resume;

Resumes a paused animation by resuming the animation thread.

The animation continues playing from the current frame.

procedure Step;

Moves a stopped animation forward by one frame.

If the current frame is the last frame of the animation, then the first frame is shown and CurrentLoop is incremented.

procedure Back;

Moves a stopped animation backward by one frame.

If the current frame is the first frame of the animation, the command has no effect.

procedure Previous;

Moves a stopped animation back to the **first** frame of the animation.

procedure Next;

Moves a stopped animation forward to the **last** frame of the animation.

procedure Eject;

This closes the panel and frees any associated memory.

procedure Paint;

This causes the contents of the animation buffer to be painted to the screen context of the panel.

property Canvas: TCanvas;

The screen canvas of the panel, on which the animation is displayed.

property FlicSurface: TFlicSurface;

The animation buffer - a Device Independent Bitmap.

property NoPaintLoad: boolean;

When set to TRUE, the first frame of the flic is not drawn automatically when the panel is opened.

property HideFlicWindow: boolean;

Normally the frame at which the animation is stopped is displayed on the screen. If this value is set to TRUE, the animation is only displayed when it is playing.

property NotifyMessages: boolean;

Enables notification messages being sent to the window identified by CallbackHandle.

property StatusMessages: boolean;

Enables status messages begin sent to the window identified by CallbackHandle.

property Stretchable: boolean;

When set to TRUE, the animation is stretched to fit the size of the panel.
When set to FALSE, the panel auto-sizes itself to any animation that is loaded.

property AutoPlay: boolean;

When set to TRUE, the animation plays immediately upon loading.

procedure Notify(Loop, Frame, MsgNum: integer);

Notify allows an application to be notified at specific frames when an animation is playing.

Loop and Frame define the position at which the notification is to take place, and MsgNum defines the parameter to be passed via the notification message.

procedure Cancel(TopLoop, TopFrame, BottomLoop, BottomFrame: integer);

This procedure cancels any pending notification events (set with Notify) in the range BottomLoop + BottomFrame to TopLoop + TopFrame.

procedure CancelAll;

This procedure cancels all pending notification events set using Notify.

property NotifyMessageNum: word;

The message number assigned by RegisterWindowMessage for the message:

‘FlicPanel Notify'; notification message

This message number allows the WndProc() of a form (or panel etc.) to access the notification messages generated by the panel in response to Notify() settings.

property StopMessageNum: word;

The message number assigned by RegisterWindowMessage for the message:

'FlicPanel Stop'; stop message

This message number allows the WndProc() of a form (or panel etc.) to access any stop message generated by the panel.

property FrameMessageNum: word;

The message number assigned by RegisterWindowMessage for the message:

‘FlicPanel Frame; notification message

This message number allows the WndProc() of a form (or panel etc.) to access the notification messages generated by the panel whenever a flic frame is rendered.

property RedimMessageNum: word;

The message number assigned by RegisterWindowMessage for the message:

'FlicPanel Redim'; notification message

This message number allows the WndProc() of a form to access the notification messages generated by the panel in response to a dynamic re-dimensioning of the arrays containing the notification data..

property FlicHeight: integer;

Returns the height of the animation. (Read only)

property FlicWidth: integer;

Returns the width of the animation. (Read only)

property FlicFrames: integer;

Returns a word containing the number of frames in the animation. (Read only)

property DesignSpeed: AASPEED;

The original design speed of the animation, in milliseconds per frame. (Read only)

property Speed: AASPEED;

The speed, in milliseconds per frame.

property CallbackHandle: THandle;

Sets the handle of the window used to receive the status and notification messages.

property X, Y: integer;

The coordinate of the upper left corner of the window used to display the animation. X and Y are relative to the upper left corner of the client window.

property CX, CY: integer;

The x offset (cx) and y offset (cy) of the animation within the panel.

property OrgX, OrgY: integer;

The coordinate within the animation buffer displayed at the upper left corner of the panel.

property CurrentFrame: integer;

Sets or reads the current frame of the animation. Counts Pascal-style from 1.

The existing animation must be stopped, or have finished playing in order to set this property.

property CurrentLoop: integer;

Sets or reads the current frame of the animation. Counts Pascal-style from 1.

The existing animation must be stopped, or have finished playing in order to set this property.

property PlayLoops: dword;

Sets the position of the loop at which the animation ends.

property PlayFrames: dword;

Sets the position of the frame at which the animation ends.

If both PlayLoops and PlayFrames are set to zero, the animation loops indefinitely.

property PauseTime: word;

Sets the length of time to pause the animation at its end, in milliseconds.

property Sound: string;

Associates a sound with an animation. The sound is begun with the animation and plays asynchronously.

property IsOpen: boolean;

Returns TRUE if the player is open. (Read only)

property FlicLoaded: boolean;

Returns TRUE if a flic file is currently loaded. (Read only)

property LastError: integer;

Returns the code number of the last error to occur in the panel.

property Status: word;

Returns a word containing current status of the animation. The status of an animation can be:

Zero is returned if no animation is loaded.
AA_STOPPED 1
The animation is loaded and is not playing.
AA_QUEUED 2
The animation is loaded and aaPlay has been used to start the animation, but the animation cannot be started because either it is a full screen animation, or a full screen animation is playing, or both. The animation will be begun as soon as possible.
AA_PLAYING 3
The animation is playing.
AA_PAUSED 4
The animation has been paused using Pause.
AA_DONE 5
A transitory state after an animation has finished playing, but before it has been stopped.

property Version: string; (read-only)

Returns a string containing the current version of the component.

property ThreadPriority: TThreadPriority;

Allows the priority setting of the flic-playing thread to be read and set.

AAPLAY.DLL functions not implemented:

Looping for the sound mode used in the Sound property.

Script support.

Transitions In and Out.

Time In and Out.

Error codes from AAPLAY.DLL (not all are implemented):

AA_ERR_NOERROR 0
No error was detected.

AA_ERR_NOMEMORY 256
An out of memory condition was detected.

AA_ERR_BADHANDLE 257
The handle used in the last call to the player was invalid.

AA_ERR_NOTIMERS 258
A system timer could not be allocated for the animation.

AA_ERR_BADSOUND 259
The sound could not be opened.

AA_ERR_NOSCRIPT 260
The operation in error requires a script.

AA_ERR_WRITEERR 261
An error occured while saving the script.

AA_ERR_BADANIMATION 262
The animation could no be opened.

AA_ERR_BADWINDOWHANDLE 512
An invalid window handle was given to the player.

AA_ERR_WINDOWCREATE 513
An error occured when creating the animation window.

AA_ERR_DLGERROR 514
An unexpected error occured while processing a dialog box.

AA_ERR_INVALIDSTATUS 768
The function failed because the state of the animation did not allow it.

AA_ERR_BADDIBFORMAT 769
The Windows DIB file is corrupted.

AA_ERR_BADFLIFORMAT 770
The Autodesk Animator or Animator Pro file is corrupted.

AA_ERR_UNRECOGNIZEDFORMAT 771
The file is in an unrecognized format.

AA_ERR_NOSOUND 772
Sound is not supported by the player.

AA_ERR_NOTVALIDFORSCRIPTS 773
The request is not allowed for scripts, only for animations.

AA_ERR_INVALIDFILE 774
The file handle for the animation is invalid.

AA_ERR_NOSCRIPTS 775
Scripts are not supported by the player.

AA_ERR_SPEED 1024
The speed is invalid. The speed must be from 19 to 1000 milliseconds.

AA_ERR_LOOPS 1025
The loops are invalid. The number of loops may range from 0 to 999.

AA_ERR_RPTSOUND 1026
The number of sound repetitions are invalid. This value may range from 0 to 1000.

AA_ERR_PAUSE 1027
The time to pause at the end of the animation is invalid. This value may range from 0 to 50000 milliseconds.

AA_ERR_TRANSIN 1028
The starting transition is invalid. Transitions must be one of the codes described above.

AA_ERR_TIMEIN 1029
The duration of the starting transition is invalid. It ranges from 250 to 10000 milliseconds.

AA_ERR_TRANSOUT 1030
The ending transition is invalid. It must be one of the codes described above.

AA_ERR_TIMEOUT 1031
The time of the ending transition is invalid. It may rang from 250 to 10000 milliseconds.

AA_ERR_DELAYSND 1032
The delay of the sound from the start of the animation is invalid. It may range from -3000000 to 3000000 milliseconds.

AA_ERR_INVALIDTYPE 1033
The type parameter to aaGetParm, aaSetParm or aaSetParmIndirect is invalid.

AA_ERR_DUPLICATENOTIFY 1280
An attempt was made to add a duplicate notification.

AA_ERR_NOSWITCH 1536
A script line is missing an option switch, or the switch is invalid.

AA_ERR_PARSELOOPS 1537
The number of loops on a script line is invalid.

AA_ERR_PARSESPEED 1538
The speed on a script line is invalid.

AA_ERR_BADRPTSOUND 1540
The number of sound repetitions on a script line is invalid.

AA_ERR_PARSEPAUSE 1541
The pause at the end of an animation on a script line is invalid.

AA_ERR_PARSETRANS 1542
A transition value is invalid.

AA_ERR_PARSEDELAYSND 1543
The delay of a sound from the beginning of the animation is invalid.