DirectX components for Delphi 3, 4

DelphiX

1998.10.05 version

Hiroyuki Hori / 堀 浩行

E-Mail: hori@ingjapan.ne.jp
Homepage: http://www.ingjapan.ne.jp/hori/

What is DelphiX?

DelphiX is a component collection to use DirectX with Borland Delphi3 easily.

These were required to be developed with C and C++ though new API named DirectX joined Windows95 . In it, it was pitiless, and I made the component collection for Delphi . Then, please enjoy the Delphi and DelphiX lives.

Contents

Installation

The DirectX5 run time is installed. And, Install_for?.exe in the Bin folder of DelphiX is executed.

Offered file

Readme.txt	Manual in English
Readmej.txt	Manual in Japanese

Bin\	
Install_for3.exe	installer for Delphi 3
Install_for4.exe	installer for Delphi 4

FFEffects\	Effect collections of force feedback
Help\	Help file
Samples\	Sample program

Source\	
DAnim.pas	DirectAnimation headers
DelphiX_for3.dcr	Resource of component for Delphi3
DelphiX_for3.dpk	Source of DelphiX package for Delphi3
DelphiX_for3.res	Resource of DelphiX package for Delphi3
DelphiX_for4.dcr	Resource of component for Delphi4
DelphiX_for4.dpk	Source of DelphiX package for Delphi4
DelphiX_for4.res	Resource of DelphiX package for Delphi4
DelphiXinc.inc	Include file which sets compiler
DIB.pas	DIB relation
DirectX.pas	DirectX headers
DShow.pas	DirectShow headers
DXClass.pas	Component, class, and routine used with DelphiX
DXConsts.pas	Character-string constant
DXDIBEdit.pas	TDIB property editor
DXDraws.pas	DirectDraw relation
DXETable.pas	Error character string
DXFFBEdit.pas	TForceFeedbackEffects property editor
DXGUIDEdit.pas	GUID property editor
DXInput.pas	Input relation
DXInptEdit.pas	TDXInput component editor
DXPlay.pas	DirectPlay relation
DXPlayFm.pas	DirectPlay connection window
DXRender.pas	Software polygon renderer
DXReg.pas	Component registration unit
DXSounds.pas	DirectSound relation
DXSprite.pas	Sprite relation
DXWaveEdit.pas	TForceFeedbackEffects property editor
Wave.pas	Wave relation

When you distribute the DelphiX use application program

When software which uses DelphiX is distributed,　it is glad when the following copyright displays can be had to be put on the manual.

Title	DelphiX
Copyright	Copyright(C) 1996-1998 Hiroyuki Hori
Address	http://www.ingjapan.ne.jp/hori/

Address of thanks

Thank you to the following people when DelphiX is made.

I was going to refer to the source and the sample of Mr. Ryo Shi-3z's Direct3Z when I mounted 3D function.

The picture of the Shoot sample was gotten from Mr. Masak Urazaki.

The 3d_shadow and 3d_sprite sample was gotten from AndyGFX.

We wish to express our gratitude to you of FDelphi which draws the voice of the advice and the encouragement.

Copyright

As for each file of DelphiX,　the copyright belongs to the author,
and is treated as free software excluding the DXHeader folder,　the FFEffects folder,　and the Samples folder.

DelphiX is used and the copyright of the made software is a thing of the author of
the software The author of DelphiX assumes the one without right and the obligation of it.

Please go without changing the content of the archive concerning the re-distribution .
Care neither addition nor to distribute another file again .
Please inform the undermentioned mail address because it is good after the thing if it is possible to do.

Moreover,　please use the result by having used this software without permission etc.
in user's responsibility because the author cannot assume the responsibility.

Historys

1998/10/05 ver 1998.10.05

- DelphiX was made DirectX6 exclusive use.
- Delphi4 was made main in the platform.
- When DelphiX is used with Delphi3, a part of a new function cannot be used.
- When DelphiX was used with Delphi4, the default argument and the over loading were used.
- The clipboard operation can have been done in the TDIB property editor.
- The icon file etc. can have been read in the TDIB property editor.
- doDrawPrimitive and doTexture of the TDXDraw.Options property were deleted. As a result, when the project of old DelphiX is read, the Options property is initialized.
- When the form to which TDXDraw was given a ride to by Windows 98 was moved, it was corrected that garbage remained.
- The TDXDraw.UpdatePalette method was always synchronized with refreshing the display.
- It was possible to right and left and upper and lower reverse by the Draw method of TDirectDrawSurface.
- It was corrected to cut the edge by the DrawAdd method of TDirectDrawSurface etc.
- The subtraction synthesis versions of the DrawAdd method and the DrawRotate method of TDirectDrawSurface etc. were added.
- The FillRectAdd method of doing the fade-in fade-out to TDirectDrawSurface more than 16 bit color, the FillRectSub method, and the FillRectAlpha method were added.
- The GammaControl property which adjusted the gamma to TDirectDrawSurface was added.
- When a new image was set in TDXImageList when designing, the color of the pixel on the left was automatically set in a transparent color.
- The mouse was supported with TDXInput. The BindInputStates property was added to the TCustomInput object as the mouse had been supported.
- The BindInputStates property was added to the TCustomInput object as the mouse had been supported.

1998/06/03 ver 1998.06.03

- The ALink key word and the tutorial were added to the help file.
- The sample (Samples\Graphic\DynImageList folder) which dynamically added the content of TDXImageList was added.
- Because PictEdit.pas might compete with other components, the name was changed into DXPictEdit.pas.
- The method of the version where DrawRotateXXX was not rotated was added.
- The DrawWaveXxxx method of transforming X direction crimp was added.
- The bug that 24 bits TBitmap were not able to do in Assign was corrected to TDIB.
- The bug of the DirectX header was corrected.

1998/05/28 ver 1998.05.28

- When you tried to access the pixel data compressed with TDIB, it was corrected the generation of the EAccessViolation exception.
- It was corrected that the TBitmapFileHeader.bfSize field of preservation in the bit map file with TDIB was an illegal value.
- The Mirror method the under's reversing right and left and reversing to TDIB was added.
- The function equal with the ConvertBitCount method was absorbed to the BitCount property of TDIB, and the ConvertBitCount method was deleted.
- The amount of the use of having GDI and the resource where HPalette was shared with TDIB was suppressed.
- The algorithm of the pfGetxxx function of the DIB unit was changed.
- Optimization of software polygon renderer
- The surface which was not the size of the involution of two can have been displayed by the DrawRotateXXX method.

- The connection to which the diamond log was not displayed with TDXPlay was enabled.
- The bug of the operation when the DXG file was read when the application was executed with TDXImageList was corrected.
- It was corrected that there was a bug in how to calculate the value of X property when TDXInput.Joystick was controlled by the JoyGetPosEx function, Y property, and Z property.
- The connection to which the diamond log was not displayed with TDXPlay was enabled.

1998/05/12 ver 1998.05.18

- The clipping was done by the TDirectDrawSurface.StretchDraw method and the TPictureCollectionItem.StretchDraw method.
- The collision judgment of TImageSprite can have been judged in each pixel. ->PixelCheck property addition
- Some memory leak was corrected.
- Optimization of software polygon renderer
- RLE was supported with TDIB. Moreover, the pixel data can have been secured for the memory.-> Compress, Decompress, Dormant, and FreeImage method addition
- The method such as shadeing off to TDIB of a special effect was added.
- The pixel format was checked with TDIB at 24 bit color.
- It was stopped to have converted TBitmap into TDIB automatically when TPicture with TBitmap in TPictureCollectionItem.Picture was substituted by making a special property editor.
- The AutoUpdate property which specified whether to call the Update method in TAudioStream by the automatic operation was added.
- The IsHost property by which it was returned to TDXPlay whether a present computer was ahost was added.
- When the TDXPlay connection diamond log of DelphiX English version was displayed in aJapanese environment, it was corrected to misconvert.

1998/05/02 ver 1998.05.02
1998/04/29 ver 1998.04.29
1998/04/24 ver 1998.04.24
1998/04/18 ver 1998.04.18
1998/04/02 ver 1998.04.02
1998/03/25 ver 1998.03.25
1998/03/10 ver 1.05a
1998/02/28 ver 1.05
1998/02/06 ver 1.04a
1998/01/31 ver 1.04
1998/01/22 ver 1.03a
1998/01/02 ver 1.03
1997/12/27 ver 1.02
1997/12/14 ver 1.01
1997/12/02 ver 1.00e
1997/11/25 ver 1.00d
1997/11/20 ver 1.00c
1997/11/17 ver 1.00b
1997/11/11 ver 1.00a
1997/11/06 ver 1.00
1997/08/02 ver 1.00 b3b
1997/07/14 ver 1.00 b3a
1997/06/07 ver 1.00 b3
1997/03/03 ver 1.00 b2
1997/02/22 ver 1.00 b1

1997/02/06 Ver 1.00 b

- DirectX was parodied and the name was changed to DelphiX.

1996/08/22 ver 0.00

- The first prototype completion
- DirectDraw Component and naming

Unit of DelphiX

The unit included in DelphiX is shown below.

DIB unit

DXClass unit

DXDraws unit

DXInput unit

DXPlay unit

DXSounds unit

DXSprite unit

Wave unit

DIB unit

The class concerning DIB is described in the DIB unit .
The following item is declared to the DIB unit.

Components

TDXDIB

TDXPaintBox

Objects

TDIB

Types

TDIBPixelFormat

TPaletteEntries

TRGBQuads

Routines

GreyscaleColorTable

MakeDIBPixelFormat

MakeDIBPixelFormatMask

pfGetBValue

pfGetGValue

pfGetRGB

pfGetRValue

pfRGB

TDXDIB

Hierarchy Properties

Unit

DIB

Description

The TDXDIB component maintains DIB. DIB property is used to access DIB.

Hierarchy

TComponent

TCustomDXDIB

TDXDIB properties

TDXDIB Lenqend

In TDXDIB

- 🟩 DIB

TDXDIB.DIB

TDXDIB

```
property DIB: TDIB;
```

Description

It is TDIB object.

TDXPaintBox

Hierarchy Properties Methods Example

Unit

DIB

Description

The TDXPaintBox component is DIB version of TImage.

DIB property is used to describe. To display DIB on the screen, the Paint method is called.

Hierarchy

TGraphicControl

TCustomDXPaint

TDXPaintBox properties

TDXPaintBox Lengend

In TDXPaintBox

- 🟩 AutoStretch
- 🟩 Center
- 🟩 DIB
- 🟩 Stretch

TDXPaintBox.AutoStretch

TDXPaintBox

```
property AutoStretch: Boolean;
```

Description

The image whether is reduced and displayed automatically when the image cannot finish being installed in the control are specified.

TDXPaintBox.Center

TDXPaintBox

```
property Center: Boolean;
```

Description

It is specified whether to display the image at the center of the control.

TDXPaintBox.DIB

TDXPaintBox Example

```
property DIB: TDIB;
```

Description

It is TDIB object . When the image is changed,   the change in the image is reflected in the screen by the Paint method.

TDXPaintBox.Stretch

TDXPaintBox

```
property Stretch: Boolean;
```

Description

It is specified whether to match the image to the size of the screen and to do the expansion reduction display.

TDXPaintBox methods

TDXPaintBox Lenqend

In TDXPaintBox

Paint

TDXPaintBox.Paint

TDXPaintBox   Example

```
procedure Paint;
```

Description

DIB is displayed on the screen.

Example of TDXPaintBox

The screen changed into an invalid channel with the television is displayed.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  DXPaintBox1.Stretch:= True;

  DXPaintBox1.DIB.ColorTable := GreyscaleColorTable;
  DXPaintBox1.DIB.SetSize(DXPaintBox1.Width div 4, DXPaintBox1.Height div 4,
8);
end;

procedure TForm1.Timer1Timer(Sender: TObject);
var
  x, y: Integer;
  P: ^TByteArray;
begin
  for y:=0 to DXPaintBox1.DIB.Height-1 do
  begin
    P := DXPaintBox1.DIB.ScanLine[y];
    for x:=0 to DXPaintBox1.DIB.Width-1 do
      P[x] := Random(200);
  end;

  DXPaintBox1.Paint;
end;
```

TDIB

Hierarchy Properties Methods Example

Unit

DIB

Description

The TDIB object maintains DIB(Device independent bitmap). The ScanLine property and the PBits property are used to access the pixel data of DIB directly.

The palette can be set with the ColorTable property. To reflect the change of the palette in the screen, the UpdatePalette method is called.

Hierarchy

TGraphic

TDIB properties

TDIB Lengend

In TDIB

BitCount

▶ BitmapInfo

▶ BitmapInfoSize

▶ Canvas

ColorTable

▶ Handle

Height

▶ NextLine

▶ NowPixelFormat

Palette

▶ PaletteCount

▶ PBits

PixelFormat

Pixels

▶ ScanLine

▶ Size

▶ TopPBits

Width

WidthBytes

TDIB.BitCount

TDIB

```
property BitCount: Integer;
```

Description

It is a number of DIB of bits for each pixel. Width Height The SetSize method is used to set BitCount at the same time.

When this property is set, the number of colors of images is converted.

TDIB.BitmapInfo

TDIB

```
property BitmapInfo: PBitmapInfo;
```

Description

It is a pointer to TBitmapInfo structural body . It is nil that the Size property is 0.

Never change this contents . The adverse effect goes out when other TDIB objects and the pixel data are shared.

TDIB.BitmapInfoSize

TDIB

```
property BitmapInfoSize: Integer;
```

Description

It is a size of the BitmapInfo property.

TDIB.Canvas

TDIB

```
property Canvas: TCanvas;
```

Description

It is a canvas.

TDIB.ColorTable

TDIB

```
property ColorTable: TRGBQuads;
```

Description

The palette can be set by using the ColorTable property . To reflect the change,   the UpdatePalette method is called.

TDIB.Handle

TDIB

```
property Handle: HBitmap;
```

Description

It is Bitmap handle.   This is a thing which the CreateDIBSection function returned.

TDIB.Height

TDIB

```
property Height: Integer;
```

Description

It is height of DIB. The SetSize method is used to set Width, Height, and BitCount at the same time.

TDIB.NextLine

TDIB Example

```
property NextLine: Integer;
```

Description

It is an offset value to the scanning line of one below . It is in case of negative.

Example of TDIB.NextLine and TopPBits

The palette number specified for coordinates specified by X and Y in eight bits DIB by C is set.

```
  PByte(Integer(DIB.TopBits)+Y*NextLine+X))^ := C;
```

TDIB.NowPixelFormat

TDIB

```
property NowPixelFormat: TDIBPixelFormat;
```

Description

It is a present pixel format.

TDIB.Palette

TDIB

```
property Palette: HPalette;
```

Description

It is Handle of the palette.

TDIB.PaletteCount

TDIB

```
property PaletteCount: Integer;
```

Description

It is a number of entries of palettes.

TDIB.PBits

TDIB

```
property PBits: Pointer;
```

Description

It is a pointer to the pixel data.

TDIB.PixelFormat

TDIB

```
property PixelFormat: TDIBPixelFormat;
```

Description

It is a pixel format when DIB is made. It is effective at 16 bits, 24 bits, and 32 bits.

TDIB.Pixels

TDIB

```
property Pixels[X, Y: Integer]: Longint;
```

Description

The pixel of DIB can be acquired,　and be set.

TDIB.ScanLine

TDIB

```
property ScanLine[Y: Integer]: Pointer;
```

Description

It is a pointer to the pixel data of the scanning line specified by Y of DIB.

TDIB.Size

TDIB

```
property Size: Integer;
```

Description

It is a size of the pixel data of DIB.

TDIB.TopPBits

TDIB

```
property TopPBits: Pointer;
```

Description

It is a pointer to the pixel data on the left of DIB. This is used together with the NextLine property. This is the same as ScanLine[0].

TDIB.Width

TDIB

```
property Width: Integer;
```

Description

It is Width of DIB . The SetSize method is used to set Width,   Height,   and BitCount at the same time.

TDIB.WidthBytes

TDIB

```
property WidthBytes: Integer;
```

Description

The number of bytes used by one line of DIB is returned . This is always a multiple of four .
The NextLine property is used to know the offset value to the scanning line of one below.

TDIB methods

TDIB   Lenqend

In TDIB

- Assign
- Blur
- Clear
- Compress
- Create
- Decompress
- Destroy
- Dormant
- FreeImage
- Greyscale
- LoadFromClipboardFormat
- LoadFromFile
- LoadFromStream
- Mirror
- Negative
- SaveToClipboardFormat
- SaveToFile
- SaveToStream
- SetSize
- UpdatePalette

TDIB.Assign

TDIB

```
procedure Assign(Source: TPersistent);
```

Description

The Assign method copies the object compatible with TDIB. The object at present compatible is TDIB, TGraphic,　and TPicture object.

When the TDIB object is specified for the copy origin,　the resource is shared.

TDIB.Blur

TDIB

```
procedure Blur(ABitCount: Integer; Radius: Integer);
```

Description

The image is shadeed off . The OnProgress event is generated because it takes time to this method for processing.

If eight or less is specified for the ABitCount argument, the result becomes a grayscale image.

Argument	Explanation
ABitCount	Number of bits of results
Radius	A square radius by which the average is calculated . The image is shadeed off the specification of a large value strongly.

TDIB.Clear

TDIB

```
procedure Clear;
```

Description

DIB is abandoned.

TDIB.Compress

TDIB

```
procedure Compress;
```

Description

DIB being maintained now is compressed in RLE . To restore the compressed data, the Decompress method is called.

When the access of DIB to the pixel data is needed, the Decompress method is automatically called . However, the Draw method is an exception.

TDIB.Create

TDIB

```
constructor Create;
```

Description

The TDIB object is made.

TDIB.Decompress

TDIB

```
procedure Decompress;
```

Description

If DIB being maintained now is compressed,   it is restored.

TDIB.Destroy

TDIB

```
destructor Destroy;
```

Description

The TDIB object is abandoned . DIB maintained at this time is abandoned.

TDIB.Dormant

TDIB

```
procedure Dormant;
```

Description

DIB is secured for not the GDI resource but the memory . To secure DIB for the GDI resource oppositely, the FreeImage method is called.

The GDI resource can be saved by calling this method.

TDIB.FreeImage

TDIB

```
procedure FreeImage;
```

Description

DIB is secured for not the memory but the GDI resource. To secure the memory DIB oppositely, the Dormant method is called.

If DIB is compressed, it is automatically restored.

TDIB.Greyscale

TDIB

```
procedure Greyscale(ABitCount: Integer);
```

Description

DIB is converted into the grayscale image.

Argument	Explanation
ABitCount	Number of bits of results

TDIB.LoadFromClipboardFormat

TDIB

```
procedure LoadFromClipboardFormat(AFormat: Word; AData: THandle; APalette:
  HPALETTE);
```

Description

DIB is read from the clipboard.

TDIB.LoadFromFile

TDIB

```
procedure LoadFromFile(const FileName: string);
```

Description

DIB is read and the file is read . The corresponding type is RGB and OS/2.

TDIB.LoadFromStream

TDIB

```
procedure LoadFromStream(Stream: TStream);
```

Description

DIB is read from the stream . The corresponding type is RGB and OS/2.

TDIB.Mirror

TDIB

```
procedure Mirror(MirrorX, MirrorY: Boolean);
```

Description

The direction of the image is reversed.

Argument	Explanation
MirrorX	It is specified whether to reverse the direction of the image according to horizontal direction.
MirrorX	It is specified whether to reverse the direction of the image according to the vertical direction.

TDIB.Negative

TDIB

```
procedure Negative;
```

Description

The color of DIB is reversed . The palette is reversed below 256 colors,   and the pixel data is reversed besides.

TDIB.SaveToClipboardFormat

TDIB

```
procedure SaveToClipboardFormat(var AFormat: Word; var AData: THandle; var
  APalette: HPALETTE);
```

Description

DIB is saved to the clipboard.

TDIB.SaveToFile

TDIB

```
procedure SaveToFile(const FileName: string);
```

Description

DIB is saved to the file.

TDIB.SaveToStream

TDIB

```
procedure SaveToStream(Stream: TStream);
```

Description

DIB is saved to the stream.

TDIB.SetSize

TDIB

```
procedure SetSize(AWidth, AHeight, ABitCount: Integer);
```

Description

Width, height, and the number of bits of DIB are set . DIB before is lost.
Please set the PixelFormat property correctly at 16 bits, 24 bits, and 32 bits.

TDIB.UpdatePalette

TDIB

```
procedure UpdatePalette;
```

Description

The change of the palette is reflected in the screen.

See also

ColorTable

TDIBPixelFormat type

Unit

DIB

Declaration

```
TDIBPixelFormat = record
  RBitMask, GBitMask, BBitMask: Integer;
  RBitCount, GBitCount, BBitCount: Integer;
  RShift, GShift, BShift: Integer;
end;
```

Description

The TDIBPixelFormat type defines the pixel format of DIB . The MakeDIBPixelFormat function or the MakeDIBPixelFormatMask function is used to generate this type.

Identifier	Meaning
RBitMask, GBitMask, BBitMask	Bit mask of element each RGB
RBitCount, GBitCount, BBitCount	Number of bits a pixel of elements each RGB
RShift, GShift	Number of right shifts to correct color to eight bits
BShift	Number of left shifts to correct color to eight bits

TPaletteEntries type

Unit

DIB

Declaration

```
TPaletteEntries = array[0..255] of TPaletteEntry;
```

Description

The TPaletteEntries type is an array with 256 elements of TPaletteEntry.

TRGBQuads type

Unit

DIB

Declaration

```
TRGBQuads = array[0..255] of TRGBQuad;
```

Description

The TRGBQuads type is a color table of TDIB.
This type is used as a color table of the DelphiX standard.

GreyscaleColorTable function

Unit

DIB

Declaration

```
function GreyscaleColorTable: TRGBQuads;
```

Description

The color table of the grayscale is made.

MakeDIBPixelFormat function

Unit

DIB

Declaration

```
function MakeDIBPixelFormat(RBitCount, GBitCount, BBitCount: Integer):
  TDIBPixelFormat;
```

Description

The pixel format for which DIB is specified is made .

For instance,　when the RGB:565 format is made,　it is assumed *MakeDIBPixelFormat*(5, 6, 5).

Identifier	Meaning
RBitCount, GBitCount, BBitCount	Number of bits a pixel of elements each RGB

MakeDIBPixelFormatMask function

Unit

DIB

Declaration

```
function MakeDIBPixelFormat(RBitMask, GBitMask, BBitMask: Integer):
  TDIBPixelFormat;
```

Description

The pixel format for which DIB is specified is made from the mask.

For instance,　when the RGB:565 format is made,　it is assumed *MakeDIBPixelFormatMask*($FF0000, $00FF00, $0000FF).

Argument	Explanation
RBiitMask, GBitMask, BBitMask	Bit mask of element each RGB1

pfGetBValue function

Unit

DIB

Declaration

```
function pfGetBValue(const PixelFormat: TDIBPixelFormat; Color: Integer):
  Byte;
```

Description

B element of the color of the specified pixel format is pulled out.

pfGetGValue function

Unit

DIB

Declaration

```
function pfGetGValue(const PixelFormat: TDIBPixelFormat; Color: Integer):
  Byte;
```

Description

G element of the color of the specified pixel format is pulled out.

pfGetRGB function

Unit

DIB

Declaration

```
procedure pfGetRGB(const PixelFormat: TDIBPixelFormat; Color: Integer; var R,
  G, B: Byte);
```

Description

The color of the specified pixel format is decomposed into RGB.

pfGetRValue function

Unit
DIB

Declaration

```
function pfGetRValue(const PixelFormat: TDIBPixelFormat; Color: Integer):
  Byte;
```

Description
R element of the color of the specified pixel format is pulled out.

pfRGB function

Unit

DIB

Declaration

```
function pfRGB(const PixelFormat: TDIBPixelFormat; R, G, B: Byte): Integer;
```

Description

The color of the specified pixel format is made.

DXClass unit

The routine etc. used with DelphiX are declared to the DXClass unit.
The following item is declared to the DXClass unit.

Components

TDXForm

TDXTimer

Objects

TDirectX

TDirectXDriver

TDirectXDrivers

Routines

Cos256

Max

Min

OverlapRect

PointInRect

RectInRect

Sin256

TDXForm

Hierarchy Methods

Unit

DXClass

Description

The TDXForm component is a thing to optimize TForm for DelphiX.

See also

Form is transformed to TDXForm.

Hierarchy

TForm

TDXForm methods

TDXForm Lengend

In TDXForm

- RestoreWindow
- StoreWindow

TDXForm.RestoreWindow

TDXForm

```
procedure RestoreWindow;
```

Description

The state of the window preserved by the StoreWindow method is restored.

TDXForm.StoreWindow

TDXForm

```
procedure StoreWindow;
```

Description

The state of the window is preserved.

TDXTimer

Hierarchy Properties Events

Unit

DXClass

Description

The TDXTimer component is more high-speed timer than the TTimer component.
Because this component uses the Application.OnIdle event, only one can be used at the same time.

Hierarchy

TComponent

TCustomDXTimer

TDXTimer properties

TDXTimer   Lengend

In TDXTimer

ActiveOnly

Enabled

FrameRate

Interval

MaxLag

TDXTimer.ActiveOnly

TDXTimer

```
property ActiveOnly: Boolean;
```

Description

This property specifies whether the application is generated of the OnTimer event only at the time of active.

TDXTimer.Enabled

TDXTimer

```
property Enabled: Boolean;
```

Description

It is specified whether the timer is effective.

TDXTimer.FrameRate

TDXTimer

```
property FrameRate: Integer;
```

Description

The frequency which generated the OnTimer event in one second in the past is returned.

TDXTimer.Interval

TDXTimer

```
property Interval: Integer;
```

Description

The frequency by which TDXTimer generates the OnTimer event in one second is specified . If 0 is specified,   the OnTimer event is generated at as fast a cycle as possible.

TDXTimer.MaxLag

TDXTimer

```
property MaxLag: Integer;
```

Description

The maximum value of the LagCount argument of the OnTimer event is specified.

TDXTimer events

TDXTimer Lenqend

In TDXTimer

OnActivate

OnDeactivate

OnTimer

TDXTimer.OnActivate

TDXTimer

```
property OnActivate: TNotifyEvent;
```

Description

It is an event generated when the application becomes active.

TDXTimer.OnDeactivate

TDXTimer

```
property OnDeactivate: TNotifyEvent;
```

Description

It is an event generated when the application becomes not active.

TDXTimer.OnTimer

TDXTimer

```
TDXTimerEvent = procedure(Sender: TObject; LagCount: Integer) of object;
property OnTimer: TDXTimerEvent;
```

Description

It is a timer event.

Argument	Explanation
Sender	Object in generation origin of event.
LagCount	Late frequency + 1

TDirectX

Hierarchy Properties

Unit

DXClass

Description

The TDirectX object is an abstraction class of the DirectX object of DelphiX .
The DXResult property is used for knowing the return value of the function of DirectX called at the end.

Hierarchy

TPersistent

TDirectX properties

TDirectX Lengend

In TDirectX

DXResult

TDirectX.DXResult

TDirectX

```
property DXResult: HRESULT;
```

Description

The return value of the function of DirectX called at the end is maintained .

When the return value is substituted for the DXResult property,　error processing according to the value is done.

TDirectXDriver

Hierarchy Properties

Unit

DXClass

Description

It is information on the driver of DirectX.
The GUID property is used for obtaining GUID of the driver.

Hierarchy

TCollectionItem

TDirectXDriver properties

TDirectXDriver   Lenqend

In TDirectXDriver

Description

DriverName

GUID

TDirectXDriver.Description

TDirectXDriver

```
property Description: string;
```

Description

It is a description of the driver.

TDirectXDriver.DriverName

TDirectXDriver

```
property DriverName: string;
```

Description

It is a name of the driver.

TDirectXDriver.GUID

TDirectXDriver

```
property GUID: PGUID;
```

Description

It is a pointer to GUID of the driver . It is at nil.

TDirectXDrivers

Hierarchy Properties

Unit

DXClass

Description

TDirectXDrivers is a list of the driver of DirectX.

Hierarchy

TCollection

TDirectXDrivers properties

TDirectXDrivers   Lenqend

In TDirectXDrivers

Count

Drivers

TDirectXDrivers.Count

TDirectXDrivers

```
property Count: Integer;
```

Description

It is a number of drivers.

TDirectXDrivers.Drivers

TDirectXDrivers

```
property Drivers[Index: Integer]: TDirectXDriver; default;
```

Description

It is a list of the driver.

Cos256 function

Unit

DXClass

Declaration

```
function Cos256(i: Integer): Double;
```

Description

The cosine at 256 cycles is calculated at high speed.

Max function

Unit

DXClass

Declaration

```
function Max(B1, B2: Integer): Integer;
```

Description

Large one of two given numbers is returned.

Min function

Unit

DXClass

Declaration

```
function Min(B1, B2: Integer): Integer;
```

Description

Small one of two given numbers is returned.

OverlapRect function

Unit
DXClass

Declaration
```
function OverlapRect(const Rect1, Rect2: TRect): Boolean;
```

Description
It is returned whether the Rect1 rectangle and the Rect2 rectangle come in succession.

PointInRect function

Unit

DXClass

Declaration

```
function PointInRect(const Point: TPoint; const Rect: TRect): Boolean;
```

Description

It is returned whether in the rectangle which the point which had been specified by the Point argument specified by the Rect argument.

RectInRect function

Unit

DXClass

Declaration

```
function RectInRect(const Rect1, Rect2: TRect): Boolean;
```

Description

It is returned whether the rectangle specified by the Rect1 argument and the rectangle specified by the Rect2 argument come in succession.

Sin256 function

Unit

DXClass

Declaration

```
function Sin256(i: Integer): Signle;
```

Description

The sign at 256 cycles is calculated at high speed.

DXDraws unit

The class concerning the DirectDraw is declared to the DXInput unit.
The following item is declared to the DXDraws unit.

Components

TDX3D

TDXDraw

TDXImageList

Objects

TDirect3DRMUserVisual

TDirect3DTexture

TDirectDraw

TDirectDrawClipper

TDirectDrawDisplay

TDirectDrawDisplayMode

TDirectDrawPalette

TDirectDrawSurface

TDirectDrawSurfaceCanvas

TPictureCollection

TPictureCollectionItem

TDX3D

Hierarchy Properties Methods Events

Unit

DXDraws

Description

The TDX3D component is a component to use Direct3D easily . This is left only because of interchangeability with the version before . Please use the TDXDraw component because this extra uses the video memory .

Please specify the DXDraw property because this operates while cooperating with the TDXDraw component.

Hierarchy

TComponent

TCustomDX3D

TDX3D properties

TDX3D Lenqend

In TDX3D

AutoSize

Camera

CanDraw

D3D

D3D2

D3D3

D3DDevice

D3DDevice2

D3DDevice3

D3DRM

D3DRM2

D3DRMDevice

D3DRMDevice2

DXDraw

Initialized

NowOptions

Options

Scene

Surface

SurfaceHeight

SurfaceWidth

Viewport

ZBuffer

TDX3D.AutoSize

TDX3D

```
property AutoSize: Boolean;
```

Description

Whether the size of Surface also changes when the size of the Surface of the TDXDraw component specified with the DXDraw property is changed is specified.

TDX3D.Camera

TDX3D

```
property Camera: IDirect3DRMFrame;
```

Description

It is a camera frame of Direct3DRM.

TDX3D.CanDraw

TDX3D

```
property CanDraw: Boolean;
```

Description

The CanDraw property can be described now or is returned . Please check this property when describing.

TDX3D.D3D

TDX3D

```
property D3D: IDirect3D;
```

Description

It is IDirect3D interface.

TDX3D.D3D2

TDX3D

```
property D3D2: IDirect3D2;
```

Description

It is IDirect3D2 interface.

TDX3D.D3D3

TDX3D

```
property D3D3: IDirect3D3;
```

Description

It is IDirect3D3 interface.

TDX3D.D3DDevice

TDX3D

```
property D3DDevice: IDirect3DDevice;
```

Description

It is IDirect3DDevice interface.

TDX3D.D3DDevice2

TDX3D

```
property D3DDevice2: IDirect3DDevice2;
```

Description

It is IDirect3DDevice2 interface.

TDX3D.D3DDevice3

TDX3D

```
property D3DDevice3: IDirect3DDevice3;
```

Description

It is IDirect3DDevice3 interface.

TDX3D.D3DRM

TDX3D

```
property D3DRM: IDirect3DRM;
```

Description

It is IDirect3DRM interface.

TDX3D.D3DRM2

TDX3D

```
property D3DRM2: IDirect3DRM2;
```

Description

It is IDirect3DRM2 interface.

TDX3D.D3DRMDevice

TDX3D

```
property D3DRMDevice: IDirect3DRMDevice;
```

Description

It is IDirect3DRMDevice interface.

TDX3D.D3DRMDevice2

TDX3D

```
property D3DRMDevice2: IDirect3DRMDevice2;
```

Description

It is IDirect3DRMDevice2 interface.

TDX3D.DXDraw

TDX3D

```
property DXDraw: TDXDraw;
```

Description

It is TDXDraw component cooperation ahead . Please specify this property.

TDX3D.Initialized

TDX3D

```
property Initialized: Boolean;
```

Description

The TDX3D component is being initialized now or the Initialized property is returned.

TDX3D.NowOptions

TDX3D

```
property NowOptions: TDX3DOptions;
```

Description

It is setting of present of the TDX3D component . To change setting,   the Options property is set.

TDX3D.Options

TDX3D

```
type TDX3DOptions = set of (toSystemMemory, toHardware, toRetainedMode,
  toSelectDriver, toZBuffer, toRGB, toMono, toDither);
```

```
property Options: TDX3DOptions;
```

Description

How of the operation of the TDX3D component can be set by using the *Options* property . The content set in the NowOptions property according to the set content is reflected.

The following value can be specified for Options.

Identifier	Meaning
toSystemMemory	The Surface is secured for the system memory . The speed drops very much when this is specified.
toHardware	The hardware acceleration is received with Direct3D.
toRetainedMode	Direct3D Retained Mode is used.
toSelectDriver	The DirectDraw driver is selected at a full screen . To use Voodoo and PowerVR, etc. , toSelectDriver is made True.
toZBuffer	Z buffer is used.
toRGB	RGB model is used for the color model of Direct3D.
toMono	The black and white model (Or, lamp model) is used for the color model of Direct3D . If hardware can be used, the color model automatically becomes RGB model.
toDither	The dithering is enabled.

TDX3D.Scene

TDX3D

```
property Scene: IDirect3DRMFrame;
```

Description

It is a scane frame of Direct3DRM.

TDX3D.Surface

TDX3D

```
property Surface: TDirectDrawSurface;
```

Description

It is a Surface.

TDX3D.SurfaceHeight

TDX3D

```
property SurfaceHeight: Integer;
```

Description

It is height of the Surface.

TDX3D.SurfaceWidth

TDX3D

```
property SurfaceWidth: Integer;
```

Description

It is width of the Surface.

TDX3D.Viewport

TDX3D

```
property Viewport: IDirect3DRMViewport;
```

Description

It is IDirect3DRMViewport interface.

TDX3D.ZBuffer

TDX3D

```
property ZBufer: TDirectDrawSurface;
```

Description

It is Z buffer.

TDX3D methods

TDX3D Lengend

In TDX3D

Render

SetSize

TDX3D.Render

TDX3D Example

```
procedure Render;
```

Description

Rendering is done to Surface.

Example of TDX3D.Render

The image by which rendering is done is forwarded to the surface of the TDXDraw component.

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if not DXDraw1.CanDraw then Exit;
  if not DX3D1.CanDraw then Exit;

  DX3D1.Render;
  DXDraw1.Surface.Draw(0, 0, DX3D1.Surface.ClientRect, DX3D1.Surface, False);

  DXDraw1.Flip;
end;
```

TDX3D.SetSize

TDX3D

```
procedure SetSize(ASurfaceWidth, ASurfaceHeight: Integer);
```

Description

The size of the Surface is set.   The event is generated in order of OnFinalize and OnInitialize at this time.

TDX3D events

TDX3D Lenqend

In TDX3D

OnFinalize

OnInitialize

TDX3D.OnFinalize

TDX3D

```
property OnFinalize: TNotifyEvent;
```

Description

It is an event generated when the termination is done . The thing made here is put back in order.

TDX3D.OnInitialize

TDX3D

```
property OnInitialize: TNotifyEvent;
```

Description

It is an event generated when initialized . The frame is made here.

TDXDraw

Hierarchy Properties Methods Events

Unit

DXDraws

Description

The TDXDraw component is a component to treat DirectDraw and Direct3D easily.

A standard color table is specified with the DefColorTable property . Moreover, a present color table is specified with the ColorTable property . The color table is changed The content changed by the UpdatePalette method is reflected in the screen.

The Surface property is used to describe . To reflect the changed content in the screen, the Flip method is called.

To use Direct3D, do3D of the Options property is set in True . In addition, to use Retained Mode, doRetainedMode is set in True.

When the size of the Surface is changed to initialization by 3D, the OnFinalize3D event and the OnInitialize3D event are generated.

To use 3D device which cannot use the texture, doTexture is set in True.

To use Z buffer, doZBuffer is set in True.

To use the chip etc. only for 3D such as Voodoo, doSelectDriver is set in True . If two chips etc. only for 3D exist at the same time, one with a lot of functions is chosen.

Hierarchy

TCustomControl

TCustomDXDraw

TDXDraw properties

TDXDraw Lenqend

In TDXDraw

AutoInitialize

AutoSize

Camera

CanDraw

CanPaletteAnimation

Clipper

ColorTable

D3D

D3D2

D3D3

D3DDevice

D3DDevice2

D3DDevice3

D3DRM

D3DRM2

D3DRMDevice

D3DRMDevice2

DDraw

DefColorTable

Display

Driver

Initialized

NowOptions

Options

Palette

Primary

Scene

Surface

SurfaceHeight

SurfaceWidth

Viewport

ZBuffer

TDXDraw.AutoInitialize

TDXDraw

```
property AutoInitialize: Boolean;
```

Description

It is specified whether to call the Initialize method automatically when the application is started.

TDXDraw.AutoSize

TDXDraw

```
property AutoSize: Boolean;
```

Description

It is specified whether the size of the surface also changes when the size of the control is changed.

TDXDraw.Camera

TDXDraw

```
property Camera: IDirect3DRMFrame;
```

Description

It is a camera frame of Direct3DRM.

TDXDraw.CanDraw

TDXDraw

```
property CanDraw: Boolean;
```

Description

Whether it is possible to describe now is returned . Please check this property when describing.

TDXDraw.CanPaletteAnimation

TDXDraw Example

```
property CanPaletteAnimation: Boolean;
```

Description

It is returned whether to be able to do the palette animation now . Please check this property before doing the palette animation.

TDXDraw.Clipper

TDXDraw

```
property Clipper: TDirectDrawClipper;
```

Description

It is a clipper.

TDXDraw.ColorTable

TDXDraw

```
property ColorTable: TRGBQuads;
```

Description

It is a present color table (palette). To reflect the change in the screen,   the UpdatePalette method is called.

See also

DefColorTable

TDXDraw.D3D

TDXDraw

```
property D3D: IDirect3D;
```

Description

It is IDirect3D interface.

TDXDraw.D3D2

TDXDraw

```
property D3D2: IDirect3D2;
```

Description

It is IDirect3D2 interface.

TDXDraw.D3D3

TDXDraw

```
property D3D3: IDirect3D3;
```

Description

It is IDirect3D3 interface.

TDXDraw.D3DDevice

TDXDraw

```
property D3DDevice: IDirect3DDevice;
```

Description

It is IDirect3DDevice interface.

TDXDraw.D3DDevice2

TDXDraw

```
property D3DDevice2: IDirect3DDevice2;
```

Description

It is IDirect3DDevice2 interface.

TDXDraw.D3DDevice3

TDXDraw

```
property D3DDevice3: IDirect3DDevice3;
```

Description

It is IDirect3DDevice3 interface.

TDXDraw.D3DRM

TDXDraw

```
property D3DRM: IDirect3DRM;
```

Description

It is IDirect3DRM interface.

TDXDraw.D3DRM2

TDXDraw

```
property D3DRM2: IDirect3DRM2;
```

Description

It is IDirect3DRM2 interface.

TDXDraw.D3DRMDevice

TDXDraw

```
property D3DRMDevice: IDirect3DRMDevice;
```

Description

It is IDirect3DRMDevice interface.

TDXDraw.D3DRMDevice2

TDXDraw

```
property D3DRMDevice2: IDirect3DRMDevice2;
```

Description

It is IDirect3DRMDevice2 interface.

TDXDraw.DDraw

TDXDraw

```
property DDraw: TDirectDraw;
```

Description

It is TDirectDraw object.

TDXDraw.DefColorTable

TDXDraw

```
property DefColorTable: TRGBQuads;
```

Description

It is a color table of default (palette). When the surface is restored, this color table is used.

See also

ColorTable, UpdatePalette

TDXDraw.Display

TDXDraw

```
property Display: TDirectDrawDisplay;
```

Description

The display mode at the full screen mode is specified.

TDXDraw.Driver

TDXDraw

```
property Driver: PGUID;
```

Description

It is a driver of DirectDraw . When nil is specified,　the driver of default is used.

See also

Drivers

TDXDraw.Initialized

TDXDraw

```
property Initialized: Boolean;
```

Description

It is returned whether to be being initialized now.

See also

Initialize

TDXDraw.NowOptions

TDXDraw

```
property NowOptions: TDXDrawOptions;
```

Description

It is setting of present of the TDX3D component . The Options property is used to change setting.

TDXDraw.Options

TDXDraw

```
type TDXDrawOptions = set of (doFullScreen, doNoWindowChange, doAllowReboot,
  doWaitVBlank, doAllowPalette256, doSystemMemory, doStretch, doCenter,
  doFlip, do3D, doHardware, doRetainedMode, doSelectDriver, doZBuffer, doRGB,
  doMono, doDither);
property Options: TDXDrawOptions;
```

Description

How of the operation of the TDXDraw component can be set by using the Options property . The content set in the NowOptions property according to the set content is reflected.

The following value can be specified for Options property.

Identifier	Meaning
doFullScreen	The screen mode is made a full screen mode.
doNoWindowChange	The thing that DirectDraw changes the form (maximize, minimize, and restore) at the full screen mode is not permitted.
doAllowReboot	The [Ctrl+Alt+Del] key is made effective at a full screen.
doWaitVBlank	Vertical scanning lines are waited for when the Flip method is called so that the screen should not flicker.
doAllowPalette256	The palette can be used by all of the 256 colors at a full screen.
doSystemMemory	The Surface is secured for the system memory . The speed drops very much when this is specified.
doStrecth	The Surface is displayed on the screen according to the size of the control.
doCenter	The Surface is displayed at the center of the control.
doFlip	It is an effective flag only in a full screen . The double buffer ring is displayed and the content of doing Surface is displayed on the screen fast . Moreover, the size of Surface always becomes equal with the size of Primary.
do3D	Direct3D is used.
doHardware	The hardware acceleration of Direct3D is made effective.
doRetainedMode	Direct3D Retained Mode is used.
doSelectDriver	The DirectDraw driver is selected at a full screen . To use Voodoo etc. , doSelectDriver is set in True.
doZBuffer	Z buffer is used. priority levels of the number of bits of Z buffers are the order from 24 bits to 16 to 32 bits.

doRGB RGB model is used for the color model of Direct3D.

doMono The black and white model (Or, lamp model) is used for the color model of Direct3D . Being able to use hardware is a color automatically model of the hardware (It is usually RGB model).

doDither The dithering is done . The number of colors which can be pseudoexpressed to make the dithering effective even by few numbers of colors is increased.

TDXDraw.Palette

TDXDraw

```
property Palette: TDirectDrawPalette;
```

Description

It is a palette .

Please do not change the palette frequently at the window mode . Because, the resource of Windows decreases fast.

TDXDraw.Primary

TDXDraw

```
property Primary: TDirectDrawSurface;
```

Description

It is a primary surface.

The operation into which the value of the Primary.IDDSurface property changes like the Primary.Assign method and the Primary.SetSize method, etc. is a strict prohibition.

TDXDraw.Scene

TDXDraw

```
property Scene: IDirect3DRMFrame;
```

Description

It is a scane frame of Direct3DRM.

TDXDraw.Surface

TDXDraw

```
property Surface: TDirectDrawSurface;
```

Description

It is a Surface . To reflect the changed content in the screen,　the Flip method is called.

The operation into which the value of the Surface.IDDSurface property changes like the Surface.Assign method and the Surface.SetSize method,　etc. is a strict prohibition.

TDXDraw.SurfaceHeight

TDXDraw

```
property SurfaceHeight: Integer;
```

Description

It is height of the Surface.

TDXDraw.SurfaceWidth

TDXDraw

```
property SurfaceWidth: Integer;
```

Description

It is width of the Surface.

TDXDraw.Viewport

TDXDraw

```
property Viewport: IDirect3DRMViewport;
```

Description

It is IDirect3DRMViewport interface.

TDXDraw.ZBuffer

TDXDraw

```
property ZBufer: TDirectDrawSurface;
```

Description

It is Z buffer.

The operation into which the value of the ZBuffer.IDDSurface property changes like the ZBuffer.Assign method and the ZBuffer.SetSize method,   etc. is a strict prohibition.

TDXDraw methods

TDXDraw Lengend

In TDXDraw

- Drivers
- Finalize
- Flip
- Initialize
- Render
- Restore
- SetSize
- UpdatePalette

TDXDraw.Drivers

TDXDraw

```
class function Drivers: TDirectXDrivers;
```

Description

The list of the DirectDraw driver can be acquired.

See also

Driver

TDXDraw.Finalize

TDXDraw

```
procedure Finalize;
```

Description

The termination is done . At this time, the event is generated in order of OnFinalizeSurface-OnFinalize.

TDXDraw.Flip

TDXDraw

```
procedure Flip;
```

Description

The content of Surface is forwarded to the screen.

See also

Surface

TDXDraw.Initialize

TDXDraw

```
procedure Initialize;
```

Description

The TDXDraw component is initialized . At this time,   the event is generated in order of OnInitializing-OnInitialize-OnInitializeSurface-OnRestoreSurface.

TDXDraw.Render

TDXDraw

```
procedure Render;
```

Description

Rendering does 3D. At this time,　when doRetainedMode of the NowOptions property is False,　nothing is done.

TDXDraw.Restore

TDXDraw

```
procedure Restore;
```

Description

The Surface etc. are restored . At this time, the OnRestoreSurface event is generated.

TDXDraw.SetSize

TDXDraw

```
procedure SetSize(ASurfaceWidth, ASurfaceHeight: Integer);
```

Description

The size of the surface is set . However,   when goFlip is included in the NowOptions property,   nothing is done . If the size of the surface is changed,   the event is generated in order of OnFinalizeSurface-OnInitializeSurface.

TDXDraw.UpdatePalette

TDXDraw Example

```
procedure UpdatePalette;
```

Description

The ColorTable property is reflected and the change is reflected in the screen .
Please check the CanPaletteAnimation property when you do the palette animation.

TDXDraw events

TDXDraw Lengend

In TDXDraw

OnFinalize

OnFinalizeSurface

OnInitialize

OnInitializeSurface

OnInitializing

OnRestoreSurface

TDXDraw.OnFinalize

TDXDraw

```
property OnFinalize: TNotifyEvent;
```

Description

It is an event generated when the termination is done . This event has paired with the OnInitialize event.

TDXDraw.OnFinalizeSurface

TDXDraw

```
property OnFinalizeSurface: TNotifyEvent;
```

Description

It is an event generated when Surface is abandoned . This event has paired with the OnInitializeSurface event . Please abandon the texture here.

TDXDraw.OnInitialize

TDXDraw

```
property OnInitialize: TNotifyEvent;
```

Description

It is an event generated when the TDXDraw component is initialized . This event has paired with the OnFinalize event.

TDXDraw.OnInitializeSurface

TDXDraw

```
property OnInitializeSurface: TNotifyEvent;
```

Description

It is an event generated when the size of Surface is changed . This event has paired with the OnFinalizeSurface event .

Please make setting 3D device and the texture here.

TDXDraw.OnInitializing

TDXDraw

```
property OnInitializing: TNotifyEvent;
```

Description

It is an event generated when the TDXDraw component tried to be initialized.

TDXDraw.OnRestoreSurface

TDXDraw

```
property OnRestoreSurface: TNotifyEvent;
```

Description

It is an event generated when the surface is restored . Please load the image into the surface made for myself here . The TDXDraw component sets DefColorTable in a present palette before generating this event.

TDXImageList

Hierarchy Properties

Unit

DXDraws

Description

The TDXImageList component maintains the list of the image . The Surface is controlled by specifying the DXDraw property.

See also

Image is displayed with TDXImageList.

Hierarchy

TComponent

TCustomDXImageList

TDXImageList properties

TDXImageList   Lenqend

In TDXImageList

DXDraw

Items

TDXImageList.DXDraw

TDXImageList

```
property DXDraw: TDXDraw;
```

Description

The TDXDraw component is specified . When this property is specified,   the Surface is controlled.

See also

TDXDraw

TDXImageList.Items

TDXImageList

```
property Items: TPictureCollection;
```

Description

It is a list of the image.

TDirect3DRMUserVisual

Hierarchy Properties Methods

Unit

DXDraws

Description

The wrapping does the IDirect3DRMUserVisual interface to the TDirect3DRMUserVisual object .
It is necessary to override the DoRender method to use this object.

Hierarchy

TObject

TDirect3DRMUserVisual properties

TDirect3DRMUserVisual Lengend

In TDirect3DRMUserVisual

UserVisual

TDirect3DRMUserVisual.UserVisual

TDirect3DRMUserVisual

```
property UserVisual: IDirect3DRMUserVisual;
```

Description

It is IDirect3DRMUserVisual interface.

TDirect3DRMUserVisual methods

TDirect3DRMUserVisual   Lengend

In TDirect3DRMUserVisual

Create

Destroy

🟨 DoRender

TDirect3DRMUserVisual.Create

TDirect3DRMUserVisual

```
constructor Create(D3DRM: IDirect3DRM);
```

Description

The TDirect3DRMUserVisual object is made.

Argument	Explanation
D3DRM	The IDirect3DRM interface . The TDXDraw.D3DRM property and the TDX3D.D3DRM property are specified.

TDirect3DRMUserVisual.Destroy

TDirect3DRMUserVisual

```
destructor Destroy;
```

Description

The TDirect3DRMUserVisual object is abandoned.

TDirect3DRMUserVisual.DoRender

TDirect3DRMUserVisual

```
function DoRender(Reason: D3DRMUSERVISUALREASON; D3DRMDev: IDirect3DRMDevice;
  D3DRMView: IDirect3DRMViewport): HResult; virtual;
```

Description

When it is necessary to do rendering, this method is called from Direct3DRM. The user overrides this method, and does rendering.

Argument	Explanation
Reason	Reason to call DoRender method. D3DRMUSERVISUAL_CANSEE The callback function should return TRUE if the user-visual object is visible in the viewport. D3DRMUSERVISUAL_RENDER The callback function should render the user-visual object.
D3DRMDev	IDirect3DRMDevice interface
D3DRMView	IDirect3DRMView interface

TDirect3DTexture

Hierarchy Properties Methods

Unit

DXDraws

Description

The wrapping does texture of Direct3D to the TDirect3DTexture object.

Hierarchy

TObject

TDirect3DTexture properties

TDirect3DTexture Lengend

In TDirect3DTexture

Handle

Surface

Texture

TDirect3DTexture.Handle

TDirect3DTexture

```
property Handle: D3DTEXTUREHANDLE;
```

Description

It is a handle of the texture.

TDirect3DTexture.Surface

TDirect3DTexture

```
property Surface: TDirectDrawSurface;
```

Description

It is a surface of the texture.

TDirect3DTexture.Texture

TDirect3DTexture

```
property Texture: IDirect3DTexture;
```

Description

It is IDirect3DTexture interface.

TDirect3DTexture methods

TDirect3DTexture Lengend

In TDirect3DTexture

Create

Destroy

Restore

TDirect3DTexture.Create

TDirect3DTexture

```
constructor Create(Graphic: TGraphic; DXDraw: TComponent);
```

Description

The TDirect3DTexture object is made . The texture is made only after the Restore method is called.

Argument	Explanation
Graphic	Image which becomes origin of texture.
DXDraw	The TDXDraw component or the TDX3D component is specified.

TDirect3DTexture.Destroy

TDirect3DTexture

```
destructor Destroy;
```

Description

The TDirect3DTexture object is abandoned.

TDirect3DTexture.Restore

TDirect3DTexture

```
procedure Restore;
```

Description

The texture is made . Because making is done by the automatic operation, the user need not call the Restore method usually.

TDirectDraw

Hierarchy Properties Methods

Unit

DXDraws

Description

The wrapping does the IDirectDraw interface to the TDirectDraw object . The user can directly access the IDirectDraw interface by using the IDraw property.

See also

TDirectDrawClipper, TDirectDrawPalette, TDirectDrawSurface

Hierarchy

TPersistent

TDirectX

TDirectDraw properties

TDirectDraw Lenqend

In TDirectDraw

IDDraw

IDDraw4

IDraw

IDraw4

Derived from TDirectX

DXResult

TDirectDraw.IDDraw

TDirectDraw

```
property IDDraw: IDirectDraw;
```

Description

This property is used to access the IDirectDraw interface directly . Even if a point different from the IDraw property is nil, the exception is not generated.

TDirectDraw.IDDraw4

TDirectDraw

```
property IDDraw4: IDirectDraw4;
```

Description

This property is used to access the IDirectDraw interface directly . Even if a point different from the IDraw4 property is nil,   the exception is not generated.

TDirectDraw.IDraw

TDirectDraw

```
property IDraw: IDirectDraw;
```

Description

This property is used to access the IDirectDraw interface directly.

TDirectDraw.IDraw4

TDirectDraw

```
property IDraw4: IDirectDraw4;
```

Description

This property is used to access the IDirectDraw4 interface directly.

TDirectDraw methods

TDirectDraw Lengend

In TDirectDraw

Create

Destroy

Drivers

TDirectDraw.Create

TDirectDraw

```
constructor Create(GUID: PGUID);
```

Description

The TDirectDraw object is made.

Argument	Explanation
GUID	The pointer to GUID of the DirectDraw driver . When nil is specified, the driver of default is used.

See also

Drivers

TDirectDraw.Destroy

TDirectDraw

```
destructor Destroy;
```

Description

The TDirectDraw object is abandoned.

TDirectDraw.Drivers

TDirectDraw

```
class function Drivers: TDirectXDrivers;
```

Description

The list of the DirectDraw driver is returned.

TDirectDrawClipper

Hierarchy Properties Methods Example

Unit

DXDraws

Description

The wrapping does the IDirectDrawClipper interface to the TDirectDrawClipper object . The user can directly access the IDirectDrawClipper interface by using the IClipper property.

When the Surface is forwarded, the clipping can be done by using this object . To do the clipping, this object is substituted for the Clipper property of the TDirectDrawSurface object.

Hierarchy

TPersistent

TDirectX

TDirectDrawClipper properties

TDirectDrawClipper Lengend

In TDirectDrawClipper

DDraw

IClipper

IDDClipper

Derived from TDirectX

DXResult

TDirectDrawClipper.DDraw

TDirectDrawClipper

```
property DDraw: TDirectDraw;
```

Description

It is TDirectDraw object.

TDirectDrawClipper.IClipper

TDirectDrawClipper

```
property IClipper: IDirectDrawClipper;
```

Description

This property is used to access the IDirectDraw interface directly.

TDirectDrawClipper.IDDClipper

TDirectDrawClipper

```
property IDDClipper: IDirectDrawClipper;
```

Description

This property is used to access the IDirectDrawClipper interface directly . Even if a point different from the IClipper property is nil,   the exception is not generated.

TDirectDrawClipper methods

TDirectDrawClipper Lenqend

In TDirectDrawClipper

Create

Destroy

SetClipRects

TDirectDrawClipper.Create

TDirectDrawClipper

```
constructor Create(ADirectDraw: TDirectDraw);
```

Description

The TDirectDrawClipper object is made.

Argument	Explanation
ADirectDraw	TDirectDraw object

TDirectDrawClipper.Destroy

TDirectDrawClipper

```
destructor Destroy;
```

Description

The TDirectDrawClipper object is abandoned.

TDirectDrawClipper.SetClipRects

TDirectDrawClipper

```
procedure SetClipRects(const Rects: array of TRect);
```

Description

The rectangle by which the clipping is done is set.

Argument	Explanation
Rects	The rectangle is arranged by doing the clipping.

Example of TDirectDrawClipper

The image is forwarded only to on the left and the lower right of the surface.

```
var
  Clipper: TDirectDrawClipper;
  Rects: array[0..1] of TRect;
begin
  DXImageList1.Items.Find('img1').Draw(DXDraw1.Surface, 0, 0, 0);

  Clipper := TDirectDrawClipper.Create(DXDraw1.DDraw);
  try
    Rects[0] := Bounds(0, 0, DXDraw1.Surface.Width div 2,
DXDraw1.Surface.Height div 2);
    Rects[1] := Bounds(DXDraw1.Surface.Width div 2, DXDraw1.Surface.Height
div 2,
      DXDraw1.Surface.Width div 2, DXDraw1.Surface.Height div 2);

    Clipper.SetClipRects(Rects);

    DXDraw1.Surface.Clipper := Clipper;
    try
      DXImageList1.Items.Find('img2').Draw(DXDraw1.Surface, 0, 0, 0);
    finally
      DXDraw1.Surface.Clipper := nil;
    end;
  finally
    Clipper.Free;
  end;

  DXDraw1.Flip;
end;
```

TDirectDrawDisplay

Hierarchy Properties Methods

Unit

DXDraws

Description

It is a list of the display mode of DirectDraw.

Hierarchy

TPersistent

TDirectDrawDisplay properties

TDirectDrawDisplay Lenqend

In TDirectDrawDisplay

BitCount

Count

FixedBitCount

FixedRatio

FixedSize

Height

Mode

Modes

Width

TDirectDrawDisplay.BitCount

TDirectDrawDisplay

```
property BitCount: Integer;
```

Description

It is a number of bits of displays . Please use the Mode property to obtain the number of bits of present displays.

TDirectDrawDisplay.Count

TDirectDrawDisplay

```
property Count: Integer;
```

Description

It is a number of display modes which can be used.

See also

Modes

TDirectDrawDisplay.FixedBitCount

TDirectDrawDisplay

```
property FixedBitCount: Boolean;
```

Description

This property specifies whether to fix the number of bits when the display mode is changed . When False is specified, the display mode is optimized.

TDirectDrawDisplay.FixedRatio

TDirectDrawDisplay

```
property FixedRatio: Boolean;
```

Description

This property specifies whether to give priority to the screen ratio when the display mode is changed.

TDirectDrawDisplay.FixedSize

TDirectDrawDisplay

```
property FixedSize: Boolean;
```

Description

This property specifies whether to fix the size when the display mode is changed.

TDirectDrawDisplay.Height

TDirectDrawDisplay

```
property Height: Integer;
```

Description

It is height of the display . Please use the Mode property to obtain the height of a present display.

TDirectDrawDisplay.Mode

TDirectDrawDisplay

```
property Mode: TDirectDrawDisplayMode;
```

Description

It is a present display mode.

TDirectDrawDisplay.Modes

TDirectDrawDisplay

```
property Modes[Index: Integer]: TDirectDrawDisplayMode; default;
```

Description

It is a list of the display mode which can be used.

See also

Count

TDirectDrawDisplay.Width

TDirectDrawDisplay

```
property Width: Integer;
```

Description

It is width of the display . Please use the Mode property to obtain the width of a present display.

TDirectDrawDisplay methods

TDirectDrawDisplay Lenqend

In TDirectDrawDisplay

IndexOf

TDirectDrawDisplay.IndexOf

TDirectDrawDisplay

```
function IndexOf(Width, Height, BitCount: Integer): Integer;
```

Description

The index of the specified display mode is returned . When not found,   -1 is returned.

TDirectDrawDisplayMode

Hierarchy Properties

Unit

DXDraws

Description

It is a display mode of DirectDraw.

See also

TDirectDrawDisplay

Hierarchy

TCollectionItem

TDirectDrawDisplayMode properties

TDirectDrawDisplayMode Lenqend

In TDirectDrawDisplayMode

BitCount

Height

SurfaceDesc

Width

TDirectDrawDisplayMode.BitCount

TDirectDrawDisplayMode

```
property BitCount: Integer;
```

Description

It is a number of bits of display modes.

TDirectDrawDisplayMode.Height

TDirectDrawDisplayMode

```
property Height: Integer;
```

Description

It is height of the display mode.

TDirectDrawDisplayMode.SurfaceDesc

TDirectDrawDisplayMode

```
property SurfaceDesc: DDSURFACEDESC;
```

Description

It is information on the display mode.

TDirectDrawDisplayMode.Width

TDirectDrawDisplayMode

```
property Width: Integer;
```

Description

It is width of the display mode.

TDirectDrawPalette

Hierarchy Properties Methods

Unit

DXDraws

Description

The wrapping does the IDirectDrawPalette interface to the TDirectDrawPalette object . The user can directly access the IDirectDrawPalette interface by using the IPalette property.

To make the palette, the CreatePalette method is called.

Hierarchy

TPersistent

TDirectX

TDirectDrawPalette properties

TDirectDrawPalette Lengend

In TDirectDrawPalette

DDraw

Entries

IDDPalette

IPalette

Derived from TDirectX

DXResult

TDirectDrawPalette.DDraw

TDirectDrawPalette

```
property DDraw: TDirectDraw;
```

Description

It is TDirectDraw object.

TDirectDrawPalette.Entries

TDirectDrawPalette

```
property Entries[Index: Integer]: TPaletteEntry;
```

Description

It is a palette entry . Please use the GetEntries method and the SetEntries method to access at a time one or more palette entry.

TDirectDrawPalette.IDDPalette

TDirectDrawPalette

```
property IDDPalette: IDirectDrawPalette;
```

Description

This property is used to access the IDirectDrawPalette interface directly . Even if a point different from the IPalette property is nil,   the exception is not generated.

TDirectDrawPalette.IPalette

TDirectDrawPalette

```
property IPalette: IDirectDrawPalette;
```

Description

This property is used to access the IDirectDrawPalette interface directly.

TDirectDrawPalette methods

TDirectDrawPalette Lenqend

In TDirectDrawPalette

- Create
- CreatePalette
- Destroy
- GetEntries
- SetEntries

TDirectDrawPalette.Create

TDirectDrawPalette

```
constructor Create(ADirectDraw: TDirectDraw);
```

Description

The TDirectDrawPalette object is made.

Argument	Explanation
ADirectDraw	TDirectDraw object

TDirectDrawPalette.CreatePalette

TDirectDrawPalette

```
function CreatePalette(Caps: Integer; const Entries): Boolean;
```

Description

The palette is made . True is returned when succeeding.

Argument	Explanation
Caps	The flag of the made palette . The following one or the one to have harmonized in logic is specified. DDPCAPS_1BIT Indicates that the index is 1 bit. There are two entries in the color table. DDPCAPS_2BIT Indicates that the index is 2 bits. There are four entries in the color table. DDPCAPS_4BIT Indicates that the index is 4 bits. There are 16 entries in the color table. DDPCAPS_8BITENTRIES Indicates that the index refers to an 8-bit color index. This flag is valid only when used with the DDPCAPS_1BIT, DDPCAPS_2BIT, or DDPCAPS_4BIT flag, and when the target surface is in 8 bpp. Each color entry is 1 byte long and is an index to a destination surface's 8-bpp palette. DDPCAPS_8BIT ndicates that the index is 8 bits. There are 256 entries in the color table. DDPCAPS_ALLOW256 Indicates that this palette can have all 256 entries defined.
Entries	The table of the palette entry . The Byte array is specified when the DDPCAPS_8BITENTRIES flag is specified, and, besides, the TPaletteEntry array is specified.

TDirectDrawPalette.Destroy

TDirectDrawPalette

```
destructor Destroy;
```

Description

The TDirectDrawPalette object is abandoned.

TDirectDrawPalette.GetEntries

TDirectDrawPalette

```
function GetEntries(StartIndex, NumEntries: Integer; var Entries): Boolean;
```

Description

The palette is collectively acquired . True is returned when succeeding.

Argument	Explanation
StartIndex	Index of start of acquired palette entry.
NumEntries	Acquired number of palette entries.
Entries	The table of the palette entry by which the result is received . The Byte array is specified when the DDPCAPS_8BITENTRIES flag is specified for the argument of the CreatePalette method, and, besides, the TPaletteEntry array is specified.

TDirectDrawPalette.SetEntries

TDirectDrawPalette

```
function SetEntries(StartIndex, NumEntries: Integer; const Entries): Boolean;
```

Description

The palette is collectively set . True is returned when succeeding.

Argument	Explanation
StartIndex	Index of start of set palette entry.
NumEntries	Set number of palette entries
Entries	The table of the set palette entry . The Byte array is specified when the DDPCAPS_8BITENTRIES flag is specified for the argument of the CreatePalette method, and, besides, the TPaletteEntry array is specified.

TDirectDrawSurface

Hierarchy Properties Methods

Unit

DXDraws

Description

The wrapping does the IDirectDrawSurface interface to the TDirectDrawSurface object . The user can directly access the IDirectDrawSurface interface by using the ISurface property.

To set the size of the surface, the SetSize method is called .

After True is set in the SystemMemory property to secure the surface for the system memory, the SetSize method is called.

Hierarchy

TPersistent

TDirectX

TDirectDrawSurface properties

TDirectDrawSurface   Lengend

In TDirectDrawSurface

- BitCount
- Canvas
- ClientRect
- ▶ Clipper
- ▶ ColorKey
- DDraw
- GammaControl
- Height
- IDDSurface
- IDDSurface4
- ISurface
- ISurface4
- ▶ Palette
- Pixels
- SurfaceDesc
- SystemMemory
- ▶ TransparentColor
- Width

Derived from TDirectX

- DXResult

TDirectDrawSurface.BitCount

TDirectDrawSurface

```
property BitCount: Integer;
```

Description

It is a number of surfaces of bits a pixel.

TDirectDrawSurface.Canvas

TDirectDrawSurface Example

```
property Canvas: TDirectDrawSurfaceCanvas;
```

Description

It is a canvas of the surface . Please call the Release method of Canvas when you finished using the canvas.

TDirectDrawSurface.ClientRect

TDirectDrawSurface

```
property ClientRect: TRect;
```

Description

It is a rectangle of the surface . It is same as the thing returned by Rect(0, 0, Width, Height).

TDirectDrawSurface.Clipper

TDirectDrawSurface

```
property Clipper: TDirectDrawClipper;
```

Description

It is a clipper of the surface.

TDirectDrawSurface.ColorKey

TDirectDrawSurface

```
property ColorKey[Flags: Integer]: DDCOLORKEY;
```

Description

It is a color key to the surface.

Argument	Explanation
Flags	Color key flag DDCKEY_COLORSPACE Set if the structure contains a color space. Not set if the structure contains a single color key. DDCKEY_DESTBLT Set if the structure specifies a color key or color space to be used as a destination color key for blit operations. DDCKEY_DESTOVERLAY Set if the structure specifies a color key or color space to be used as a destination color key for overlay operations. DDCKEY_SRCBLT Set if the structure specifies a color key or color space to be used as a source color key for blit operations. DDCKEY_SRCOVERLAY Set if the structure specifies a color key or color space to be used as a source color key for overlay operations.

TDirectDrawSurface.DDraw

TDirectDrawSurface

```
property DDraw: TDirectDraw;
```

Description

It is TDirectDraw object.

TDirectDrawSurface.GammaControl

TDirectDrawSurface

```
property GammaControl: IDirectDrawGammaControl;
```

Description

It is a gamma control related to the surface. When the gamma control is not supported, nil is returned.

If the gamma control is used, red, green, and the output value corresponding to an input value respectively blue can be set. Doing the fade-in and the fade-out, etc. by using this more than 16 bit color becomes possible.

Attention!! The gamma control is supported only at full-screen.

TDirectDrawSurface.Height

TDirectDrawSurface

```
property Height: Integer;
```

Description

It is height of the surface.

TDirectDrawSurface.IDDSurface

TDirectDrawSurface

```
property IDDSurface: IDirectDrawSurface;
```

Description

This property is used to access the IDirectDrawSurface interface directly . Even if a point different from the ISurface property is nil,   the exception is not generated.

TDirectDrawSurface.IDDSurface4

TDirectDrawSurface

```
property IDDSurface4: IDirectDrawSurface4;
```

Description

This property is used to access the IDirectDrawSurface4 interface directly . Even if a point different from the ISurface4 property is nil,   the exception is not generated.

TDirectDrawSurface.ISurface

TDirectDrawSurface

```
property ISurface: IDirectDrawSurface;
```

Description

This property is used to access the IDirectDrawSurface interface directly.

TDirectDrawSurface.ISurface4

TDirectDrawSurface

```
property ISurface4: IDirectDrawSurface4;
```

Description

This property is used to access the IDirectDrawSurface4 interface directly.

TDirectDrawSurface.Palette

TDirectDrawSurface

```
property Palette: TDirectDrawPalette;
```

Description

It is a palette of the surface.

TDirectDrawSurface.Pixels

TDirectDrawSurface

```
property Pixels[X, Y: Integer]: Longint;
```

Description

The pixel of the surface is acquired,   and set.

TDirectDrawSurface.SurfaceDesc

TDirectDrawSurface

```
property SurfaceDesc: DDSURFACEDESC;
```

Description

It is information on the surface . Please refer to the document of DirectX for details.

TDirectDrawSurface.SystemMemory

TDirectDrawSurface

```
property SystemMemory: Boolean;
```

Description

It is specified whether to secure the surface for the system memory.

TDirectDrawSurface.TransparentColor

TDirectDrawSurface Example

```
property TransparentColor: Longint;
```

Description

It is a transparent color of the surface . Please specify the color expressed by the pixel format of the surface .

The color converted by the value acquired in the Pixels property or the ColorMatch method can be set in this property.

Example of TDirectDrawSurface.TransparentColor

When you set the color on the left of the surface in a transparent color

```
Surface.TransparentColor := Surface.Pixels[0, 0];
```

When you set clRed in a transparent color

```
Surface.TransparentColor := Surface.ColorMatch(clRed);
```

TDirectDrawSurface.Width

TDirectDrawSurface

```
property Width: Integer;
```

Description

It is width of the surface.

TDirectDrawSurface methods

TDirectDrawSurface Lengend

In TDirectDrawSurface

- Assign
- AssignTo
- Blt
- BltFast
- ColorMatch
- Create
- CreateSurface
- Destroy
- Draw
- DrawAdd
- DrawAlpha
- DrawRotate
- DrawRotateAdd
- DrawRotateAlpha
- DrawRotateSub
- DrawSub
- DrawWaveX
- DrawWaveXAdd
- DrawWaveXAlpha
- DrawWaveXSub
- Fill
- FillRect
- FillRectAdd
- FillRectAlpha
- FillRectSub
- LoadFromFile
- LoadFromGraphic
- LoadFromGraphicRect
- LoadFromStream
- Lock

Restore

SetSize

StretchDraw

UnLock

TDirectDrawSurface.Assign

TDirectDrawSurface

```
procedure Assign(Source: TPersistent);
```

Description

The Assign method allocates the object compatible with TDirectDrawSurface .
The object at present compatible is TDIB,　TGraphic,　TPicture,　and TDirectDrawSurface .
When the TDirectDrawSurface object is allocated,　the pixel data is shared.

See also

TDIB, TGraphic, TPicture

TDirectDrawSurface.AssignTo

TDirectDrawSurface

```
procedure AssignTo(Source: TPersistent);
```

Description

The AssignTo method is allocated to the object compatible with TDirectDrawSurface . The object at present compatible is TDIB.

See also

TDIB

TDirectDrawSurface.Blt

TDirectDrawSurface Example

```
function Blt(const DestRect, SrcRect: TRect; Flags: Integer; const DF:
  DDBLTFX; Source: TDirectDrawSurface): Boolean;
```

Description

The surface is described . If it is necessary, the expansion reduction is done.

Argument **Explanation**

DestRect

Rectangle description ahead

SrcRect

Rectangle of description origin

Flags

The description flag . The following one or the one to have harmonized in logic is specified.

DDBLT_ALPHADEST
Uses either the alpha information in pixel format or the alpha channel surface attached to the destination surface as the alpha channel for this blit.

DDBLT_ALPHADESTCONSTOVERRIDE
Uses the dwAlphaDestConst member of the DDBLTFX structure as the alpha channel for the destination surface for this blit.

DDBLT_ALPHADESTNEG
Indicates that the destination surface becomes more transparent as the alpha value increases (0 is opaque).

DDBLT_ALPHADESTSURFACEOVERRIDE
Uses the lpDDSAlphaDest member of the DDBLTFX structure as the alpha channel for the destination for this blit.

DDBLT_ALPHAEDGEBLEND
Uses the dwAlphaEdgeBlend member of the DDBLTFX structure as the alpha channel for the edges of the image that border the color key colors.

DDBLT_ALPHASRC
Uses either the alpha information in pixel format or the alpha channel surface attached to the source surface as the alpha channel for this blit.

DDBLT_ALPHASRCCONSTOVERRIDE
Uses the dwAlphaSrcConst member of the DDBLTFX structure as the alpha channel for the source for this blit.

DDBLT_ALPHASRCNEG
Indicates that the source surface becomes more transparent as the alpha value increases (0 is opaque).

DDBLT_ALPHASRCSURFACEOVERRIDE
Uses the lpDDSAlphaSrc member of the DDBLTFX structure as the

alpha channel for the source for this blit.

DDBLT_ASYNC
Performs this blit asynchronously through the FIFO in the order received. If no room is available in the FIFO hardware, the call fails.

DDBLT_COLORFILL
Uses the dwFillColor member of the DDBLTFX structure as the RGB color that fills the destination rectangle on the destination surface.

DDBLT_DDFX
Uses the dwDDFX member of the DDBLTFX structure to specify the effects to use for this blit.

DDBLT_DDROPS
Uses the dwDDROPS member of the DDBLTFX structure to specify the raster operations (ROPS) that are not part of the Win32 API.

DDBLT_DEPTHFILL
Uses the dwFillDepth member of the DDBLTFX structure as the depth value with which to fill the destination rectangle on the destination z-buffer surface.

DDBLT_KEYDEST
Uses the color key associated with the destination surface.

DDBLT_KEYDESTOVERRIDE
Uses the dckDestColorkey member of the DDBLTFX structure as the color key for the destination surface.

DDBLT_KEYSRC
Uses the color key associated with the source surface.

DDBLT_KEYSRCOVERRIDE
Uses the dckSrcColorkey member of the DDBLTFX structure as the color key for the source surface.

DDBLT_ROP
Uses the dwROP member of the DDBLTFX structure for the ROP for this blit. These ROPs are the same as those defined in the Win32 API.

DDBLT_ROTATIONANGLE
Uses the dwRotationAngle member of the DDBLTFX structure as the rotation angle (specified in 1/100th of a degree) for the surface.

DDBLT_WAIT
Postpones the DDERR_WASSTILLDRAWING return value if the blitter is busy, and returns as soon as the blit can be set up or another error occurs.

DDBLT_ZBUFFER
Performs a z-buffered blit using the z-buffers attached to the source and destination surfaces and the dwZBufferOpCode member of the DDBLTFX structure as the z-buffer opcode.

DDBLT_ZBUFFERDESTCONSTOVERRIDE

Performs a z-buffered blit using the dwZDestConst and dwZBufferOpCode members of the DDBLTFX structure as the z-buffer and z-buffer opcode, respectively, for the destination.

DDBLT_ZBUFFERDESTOVERRIDE

Performs a z-buffered blit using the lpDDSZBufferDest and dwZBufferOpCode members of the DDBLTFX structure as the z-buffer and z-buffer opcode, respectively, for the destination.

DDBLT_ZBUFFERSRCCONSTOVERRIDE

Performs a z-buffered blit using the dwZSrcConst and dwZBufferOpCode members of the DDBLTFX structure as the z-buffer and z-buffer opcode, respectively, for the source.

DDBLT_ZBUFFERSRCOVERRIDE

Performs a z-buffered blit using the lpDDSZBufferSrc and dwZBufferOpCode members of the DDBLTFX structure as the z-buffer and z-buffer opcode, respectively, for the source.

Source	Described surface

Example of TDirectDrawSurface.Blt

A transparent color is pulled out,　the expansion reduction is done to DXDraw1.Surface,　and Source is described.

```
var
  DF: DDBLTFX;
begin
  DF.dwsize := SizeOf(DF);
  DF.dwDDFX := 0;
  DXDraw1.Surface.Blt(DXDraw1.Surface.ClientRect, Source.ClientRect,
DDBLT_KEYSRC or DDBLT_WAIT, DF, Source);
end;
```

TDirectDrawSurface.BltFast

TDirectDrawSurface

```
function BltFast(X, Y: Integer; const SrcRect: TRect; Flags: Integer; Source:
  TDirectDrawSurface): Boolean;
```

Description

The surface is described.

Argument	Explanation
X, Y	Coordinates description ahead
SrcRect	Rectangle of description origin
Flags	The description flag . The following one or the one to have harmonized in logic is specified. DDBLTFAST_DESTCOLORKEY Specifies a transparent blit that uses the destination's color key. DDBLTFAST_NOCOLORKEY Specifies a normal copy blit with no transparency. DDBLTFAST_SRCCOLORKEY Specifies a transparent blit that uses the source's color key. DDBLTFAST_WAIT Postpones the DDERR_WASSTILLDRAWING message if the blitter is busy, and returns as soon as the blit can be set up or another error occurs.
Source	Described surface

TDirectDrawSurface.ColorMatch

TDirectDrawSurface Example

```
function ColorMatch(Col: TColor): Integer;
```

Description

This method is converted from the color specified with RGB:888 to the color expressed by the pixel format of the surface.

TDirectDrawSurface.Create

TDirectDrawSurface

```
constructor Create(ADirectDraw: TDirectDraw);
```

Description

The TDirectDrawSurface object is made.

Argument	Explanation
ADirectDraw	TDirectDraw object

TDirectDrawSurface.CreateSurface

TDirectDrawSurface

```
function CreateSurface(const SurfaceDesc: DDSURFACEDESC): Boolean; overload;
function CreateSurface(const SurfaceDesc: DDSURFACEDESC2): Boolean; overload;
```

Description

The surface is made . True is returned when succeeding.

Argument	Explanation
SurfaceDesc	DDSURFACEDESC structural body which described information on made surface

TDirectDrawSurface.Destroy

TDirectDrawSurface

```
destructor Destroy;
```

Description

The TDirectDrawSurface object is abandoned.

TDirectDrawSurface.Draw

TDirectDrawSurface

```
procedure Draw(X, Y: Integer; const SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean); overload;
procedure Draw(X, Y: Integer; Source: TDirectDrawSurface; Transparent:
  Boolean); overload;
```

Description

The surface is described . The clipping is done when beginning to see from the screen.

Argument	Explanation
X, Y	Coordinates description ahead
SrcRect	Rectangle of description origin. The forwarding origin is reversed right and left by exchanging Left of SrcRect for Right. Moreover, the forwarding origin is reversed up and down by exchanging Top of SrcRect for Bottom. It is considered that Source.ClientRect is passed to the SrcRect argument when the SrcRect argument is omitted.
Source	Described surface
Transparent	Whether a transparent color is pulled out when describing or not . When True is specified, the color set in the TransparentColor property is used for a transparent color.

TDirectDrawSurface.DrawAdd

TDirectDrawSurface

```
procedure DrawAdd(const DestRect, SrcRect: TRect; Source: TDirectDrawSurface;
  Transparent: Boolean; Alpha: Integer=255);
```

Description

The surface is synthesized in addition . However,   the addition synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
DestRect	Rectangle description ahead
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the addition synthesis is done.

TDirectDrawSurface.DrawAlpha

TDirectDrawSurface

```
procedure DrawAlpha(const DestRect, SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean; Alpha: Integer);
```

Description

The alpha-blending does the surface . However, blend is not done when the number of screen colors is below 256 colors.

Argument	Explanation
DestRect	Rectangle description ahead
SrcRect	Rectangle of source
CenterX, CenterY	The center of the surface when rotating is specified.
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TDirectDrawSurface.DrawRotate

TDirectDrawSurface

```
procedure DrawRotate(X, Y, Width, Height: Integer; SrcRect: TRect; CenterX,
  CenterY: Double; Source: TDirectDrawSurface; Transparent: Boolean; Angle:
  Integer);
```

Description

The surface is rotating described.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
CenterX, CenterY	The center of the surface when rotating is specified.
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Angle	The rotation angle is specified at 256 cycles.

TDirectDrawSurface.DrawRotateAdd

TDirectDrawSurface

```
procedure DrawRotateAdd(X, Y, Width, Height: Integer; SrcRect: TRect;
  CenterX, CenterY: Double; Source: TDirectDrawSurface; Transparent: Boolean;
  Angle: Integer; Alpha: Integer=255);
```

Description

After the surface is rotated,　the addition synthesis is done . However,　the addition synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
CenterX, CenterY	The center of the surface when rotating is specified.
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified,　the color set in the TransparentColor property is used for a transparent color.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance,　after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified,　the addition synthesis is done.

TDirectDrawSurface.DrawRotateAlpha

TDirectDrawSurface

```
procedure DrawRotateAlpha(X, Y, Width, Height: Integer; SrcRect: TRect;
  CenterX, CenterY: Double; Source: TDirectDrawSurface; Transparent: Boolean;
  Angle, Alpha: Integer);
```

Description

After the surface is rotated, the alpha-blending is done . However, the addition synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
CenterX, CenterY	The center of the surface when rotating is specified.
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TDirectDrawSurface.DrawRotateSub

TDirectDrawSurface

```
procedure DrawRotateAdd(X, Y, Width, Height: Integer; SrcRect: TRect;
  CenterX, CenterY: Double; Source: TDirectDrawSurface; Transparent: Boolean;
  Angle: Integer; Alpha: Integer=255);
```

Description

After the surface is rotated, the subtraction synthesis is done . However, the subtraction synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
CenterX, CenterY	The center of the surface when rotating is specified.
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the subtraction synthesis is done.

TDirectDrawSurface.DrawSub

TDirectDrawSurface

```
procedure DrawSub(const DestRect, SrcRect: TRect; Source: TDirectDrawSurface;
  Transparent: Boolean; Alpha: Integer=255);
```

Description

The surface is synthesized in subtraction. However, the subtraction synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
DestRect	Rectangle description ahead
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the subtraction synthesis is done.

TDirectDrawSurface.DrawWaveX

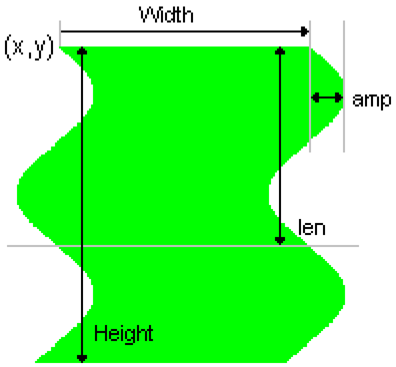
TDirectDrawSurface

```
procedure DrawWaveX(X, Y, Width, Height: Integer; SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean; amp, Len, ph: Integer);
```

Description

The shape of waves is transformed into the direction of X and the surface is described.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.



TDirectDrawSurface.DrawWaveXAdd

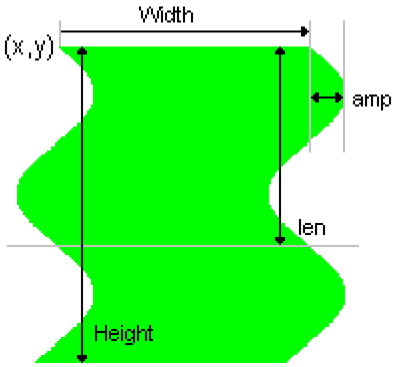
TDirectDrawSurface

```
procedure DrawWaveXAdd(X, Y, Width, Height: Integer; SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean; amp, Len, ph: Integer; Alpha:
  Integer=255);
```

Description

The surface is transformed into the direction of X in the shape of waves and the addition synthesis is done . However, the addition synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.



Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the addition synthesis is done.

TDirectDrawSurface.DrawWaveXAlpha

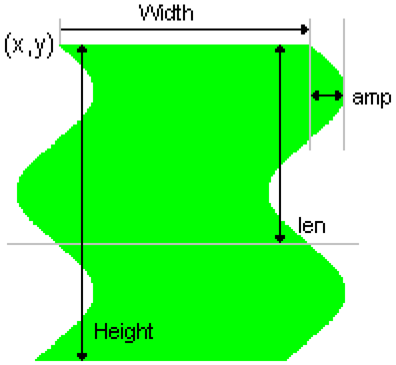
TDirectDrawSurface

```
procedure DrawWaveXAlpha(X, Y, Width, Height: Integer; SrcRect: TRect;
  Source: TDirectDrawSurface; Transparent: Boolean; amp, Len, ph, Alpha:
  Integer);
```

Description

The surface is done and the shape of waves is transformed into the direction of X and alpha-blending is done . However,   blend is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).



TDirectDrawSurface.DrawWaveXSub

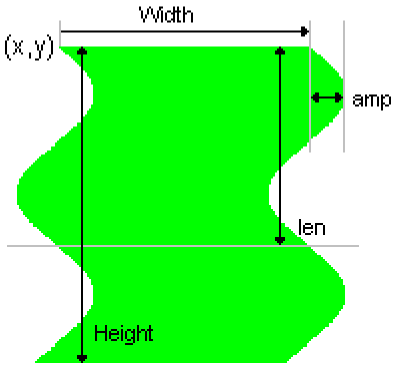
TDirectDrawSurface

```
procedure DrawWaveXSubX, Y, Width, Height: Integer; SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean; amp, Len, ph: Integer; Alpha:
  Integer=255);
```

Description

The surface is transformed into the direction of X in the shape of waves and the subtraction synthesis is done . However, the subtraction synthesis is not done when the number of screen colors is below 256 colors.

Argument	Explanation
X, Y	Coordinates of destination
Width, Height	Size of destination
SrcRect	Rectangle of source
Source	The source surface . Please secure the source surface for the system memory as much as possible.
Transparent	It is specified whether to pull out a transparent color . When True is specified, the color set in the TransparentColor property is used for a transparent color.
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the subtraction synthesis is done.



TDirectDrawSurface.Fill

TDirectDrawSurface

```
procedure Fill(DevCol: Longint);
```

Description

This method is painted out with the color for which the surface is specified . Please specify the color painted out in the form of the pixel format of the surface.

The color converted into the Col argument by the value acquired in the Pixels property or the ColorMatch method can be specified.

Argument	Explanation
DevCol	Color painted out

TDirectDrawSurface.FillRect

TDirectDrawSurface

```
procedure FillRect(Rect: TRect; DevCol: Longint);
```

Description

This method is painted out with the color for which the rectangle for which the surface is specified is specified . Please specify the color painted out in the form of the pixel format of the surface.

The color converted into the Col argument by the value acquired in the Pixels property or the ColorMatch method can be specified.

Argument	Explanation
Rect	Rectangle painted out
DevCol	Color painted out

TDirectDrawSurface.FillRectAdd

TDirectDrawSurface

```
procedure FillRectAdd(const DestRect: TRect; RGBCol: TColor);
```

Description

This method adds the color specified for the rectangle for which the surface is specified.

Argument	Explanation
Rect	Rectangle painted out
RGBCol	Added color

TDirectDrawSurface.FillRectAlpha

TDirectDrawSurface

```
procedure FillRectAlpha(const DestRect: TRect; RGBCol: TColor; Alpha:
  Integer);
```

Description

This method synthesizes the color specified for the rectangle for which the surface is specified.

Argument	Explanation
Rect	Rectangle painted out
RGBCol	Synthesized color
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TDirectDrawSurface.FillRectSub

TDirectDrawSurface

```
procedure FillRectSub(const DestRect: TRect; RGBCol: TColor);
```

Description

This method subtracts the color specified for the rectangle for which the surface is specified.

Argument	Explanation
Rect	Rectangle painted out
RGBCol	Subtracted color

TDirectDrawSurface.LoadFromFile

TDirectDrawSurface

```
procedure LoadFromFile(const FileName: string);
```

Description

The image is loaded from the file.

TDirectDrawSurface.LoadFromGraphic

TDirectDrawSurface

```
procedure LoadFromGraphic(Graphic: TGraphic);
```

Description

The image is loaded from the TGraphic object.

TDirectDrawSurface.LoadFromGraphicRect

TDirectDrawSurface

```
procedure LoadFromGraphicRect(Graphic: TGraphic; AWidth, AHeight: Integer;
  const SrcRect: TRect);
```

Description

The image of the area where Graphic was specified is loaded . The expansion reduction is done if there is a necessity.

Argument	Explanation
Graphic	Loaded image
AWidth, AHeight	The size of a new surface . It should be smaller than the size of a primary surface . When 0 is passed, the size of Graphic is used.
SrcRect	Loaded rectangle

TDirectDrawSurface.LoadFromStream

TDirectDrawSurface

```
procedure LoadFromStream(Stream: TStream);
```

Description

The bitmap is loaded from the stream . The LoadFromStream methods are no squids in TPicture only the bit map why.

TDirectDrawSurface.Lock

TDirectDrawSurface Example

```
function Lock(const Rect: TRect; var SurfaceDesc: DDSURFACEDESC): Boolean;
  overload;
function Lock(var SurfaceDesc: DDSURFACEDESC): Boolean; overload;
```

Description

The area specified with Rect of the surface is locked . True is returned when succeeding.

The user can access the pixel data directly by locking the surface . Please release the lock by the UnLock method because Windows stops when locking.

Argument	Explanation
Rect	Locked rectangle. It is assumed that ClientRect is passed when the Rect argument is omitted.
SurfaceDesc	DDSURFACEDESC structural body which receives locked information

See also

UnLock

Example of Lock and UnLock of TDirectDrawSurface

The surface is painted out with the gradation . The number of screen colors is effective only by 256 colors.

```
var
  SurfaceDesc: DDSURFACEDESC;
  x, y: Integer;
begin
  SurfaceDesc.dwSize := Sizeof(SurfaceDesc);
  if Surface.Lock(PRect(nil)^, SurfaceDesc) then
  begin
    try
      for y:=0 to Surface.Height-1 do
        for x:=0 to Surface.Width-1 do
          PByte(Integer(SurfaceDesc.lpSurface)+Y*SurfaceDesc.lPitch+X)^ :=
x+y;
    finally
      {  Without forgetting unlocking  }
      Surface.UnLock(SurfaceDesc.lpSurface);
    end;
  end;
end;
```

TDirectDrawSurface.Restore

TDirectDrawSurface

```
function Restore: Boolean;
```

Description

The surface is restored . Please load the image into the image of the surface by the LoadFromFile method etc. it is to remain being being lost even if you restore.

TDirectDrawSurface.SetSize

TDirectDrawSurface

```
procedure SetSize(AWidth, AHeight: Integer);
```

Description

The size of the surface is set . The off screen surface is made this time . Moreover,   the surface might be secured for the system memory.

Argument	Explanation
AWidth, AHeight	The size of a new surface . It should be smaller than the size of a primary surface.

See also

SystemMemory

TDirectDrawSurface.StretchDraw

TDirectDrawSurface

```
procedure StretchDraw(const DestRect, SrcRect: TRect; Source:
  TDirectDrawSurface; Transparent: Boolean=True); overload;
procedure StretchDraw(const DestRect: TRect; Source: TDirectDrawSurface;
  Transparent: Boolean=True); overload;
```

Description

The surface reduces expanding and is described . If it is necessary, the expansion reduction is done . The clipping stripes cork of this method unlike the Draw method. The TDirectDrawClipper object is used to do the clipping by all means.

Argument	Explanation
DestRect	Rectangle description ahead
SrcRect	Rectangle of description origin. Rectangle of description origin. The forwarding origin is reversed right and left by exchanging Left of SrcRect for Right. Moreover, the forwarding origin is reversed up and down by exchanging Top of SrcRect for Bottom. It is considered that Source.ClientRect is passed to the SrcRect argument when the SrcRect argument is omitted.
Source	Described surface
Transparent	Whether a transparent color is pulled out or not . When True is specified, the color set in the TransparentColor property is used for a transparent color.

TDirectDrawSurface.UnLock

TDirectDrawSurface

```
function UnLock(lpSurfaceData: Pointer): Boolean;
```

Description

The lock is released . True is returned when succeeding.

Argument	Explanation
lpSurfaceDesc	The pointer to the memory locked by the Lock method . The value of lpSurface of DDSURFACEDESC structural body is passed.

See also

Lock

TDirectDrawSurfaceCanvas

Hierarchy Methods Example

Unit

DXDraws

Description

The TDirectDrawSurfaceCanvas object is a canvas of the TDirectDrawSurface object .

Please call the Release method after finishing using the canvas. Windows stops if the Release method is not called.

See also

TDirectDrawSurface

Hierarchy

TCanvas

TDirectDrawSurfaceCanvas methods

TDirectDrawSurfaceCanvas Lenqend

In TDirectDrawSurfaceCanvas

Release

TDirectDrawSurfaceCanvas.Release

TDirectDrawSurfaceCanvas

```
procedure Release;
```

Description

The handle of the canvas is liberated.

Example of TDirectDrawSurfaceCanvas

The canvas is used and described in the surface.

```
widh DXDraw1.Surface.Canvas do
begin
  Textout(0, 0, 'DelphiX');
  Release;
end;
```

TPictureCollection

Hierarchy Properties Methods

Unit

DXDraws

Description

It is a collection of the TPicture object.

The optimization palette can be made by the MakeColorTable method . The result is stored in the ColorTable property.

See also

TPictureCollectionItem

Hierarchy

TCollection

THashCollection

TPictureCollection properties

TPictureCollection Lengend

In TPictureCollection

ColorTable

Count

Items

TPictureCollection.ColorTable

TPictureCollection

```
property ColorTable: TRGBQuads;
```

Description

It is an optimization palette made by the MakeColorTable method.

TPictureCollection.Count

TPictureCollection

```
property Count: Integer;
```

Description

The number of maintained images is returned.

See also

Items

TPictureCollection.Items

TPictureCollection

```
property Items[Index: Integer]: TPictureCollectionItem; default;
```

Description

It is a list of the maintained image.

See also

Count

TPictureCollection methods

TPictureCollection Lengend

In TPictureCollection

- Clear
- Find
- IndexOf
- LoadFromFile
- LoadFromStream
- MakeColorTable
- Restore
- SaveToFile
- SaveToStream

TPictureCollection.Clear

TPictureCollection

```
procedure Clear;
```

Description

All images being maintained now are abandoned.

TPictureCollection.Find

TPictureCollection

```
function Find(const Name: string): TPictureCollectionItem;
```

Description

The image of the specified name is returned . The hash is used and the operation is high-speed . When not found,   the exception is generated.

TPictureCollection.IndexOf

TPictureCollection

```
function IndexOf(const Name: string): Integer;
```

Description

The index of the image of the specified name is returned . The hash is used and the operation is high-speed . When not found,   -1 is returned.

TPictureCollection.LoadFromFile

TPictureCollection

```
procedure LoadFromFile(const FileName: string);
```

Description

The list of the image is read from the file.

TPictureCollection.LoadFromStream

TPictureCollection

```
procedure LoadFromStream(Stream: TStream);
```

Description

The list of the image is read from the stream.

TPictureCollection.MakeColorTable

TPictureCollection

```
procedure MakeColorTable;
```

Description

A common palette of the list of the image being maintained now is made . The made color table is stored in the ColorTable property .

The system color of Windows is sure to be included in the made color table.

TPictureCollection.Restore

TPictureCollection

```
procedure Restore;
```

Description

The image is loaded into the surface.

TPictureCollection.SaveToFile

TPictureCollection

```
procedure SaveToFile(const FileName: string);
```

Description

The list of the image being maintained now is preserved in the file.

TPictureCollection.SaveToStream

TPictureCollection

```
procedure SaveToStream(Stream: TStream);
```

Description

The list of the image being maintained now is preserved in the stream.

TPictureCollectionItem

Hierarchy Properties Methods

Unit

DXDraws

Description

It is an item of the TPictureCollection object.

The image which becomes the origin of the surface by the Picture property can be specified .

To describe, the Draw method or the StretchDraw method is called.

See also

TPictureCollection

Hierarchy

TCollectionItem

THashCollectionItem

TPictureCollectionItem properties

TPictureCollectionItem Lengend

In TPictureCollectionItem

- Height
- Initialized
- Name
- PatternHeight
- PatternRects
- PatternSurfaces
- PatternWidth
- Picture
- SkipHeight
- SkipWidth
- SystemMemory
- Transparent
- TransparentColor
- Width

TPictureCollectionItem.Height

TPictureCollectionItem

```
property Height: Integer;
```

Description

It is actual height of the pattern.

TPictureCollectionItem.Initialized

TPictureCollectionItem

```
property Initialized: Boolean;
```

Description

The surface is initialized or this property is returned.

TPictureCollectionItem.Name

TPictureCollectionItem

```
property Name: string;
```

Description

It is a name of this image.

TPictureCollectionItem.PatternHeight

TPictureCollectionItem

```
property PatternHeight: Integer;
```

Description

It is height of the pattern . It is regarded that the image can be done by specifying 0 in one pattern.

TPictureCollectionItem.PatternRects

TPictureCollectionItem Example

```
property PatternRects[Index: Integer]: TRect;
```

Description

The rectangle of the surface of the specified pattern is returned.

Rect(0, 0, 0, 0) is returned if Index is not effective.

TPictureCollectionItem.PatternSurfaces

TPictureCollectionItem Example

```
property PatternSurfaces[Index: Integer]: TRect;
```

Description

The surface of the specified pattern is returned . Please forward the rectangle returned by the PatternRects property when describing.

Nil is returned if Index is not effective.

Example of PatternRects and PatternSurfaces

Pattern 2 of Image is described.

```
var
  PatternNo: Integer;
  Image: TPictureCollectionItem;
begin
  PatternNo := 2;
  Image := DXDIBList1.Items[0];
  DXDraw1.Surface.Draw(0, 0, Image.PatternRects[PatternNo],
Image.PatternSurfaces[PatternNo], Image.Transparent);
end;
```

TPictureCollectionItem.PatternWidth

TPictureCollectionItem

```
property PatternWidth: Integer;
```

Description

It is width of the pattern . It is regarded that the image can be done by specifying 0 in one pattern.

TPictureCollectionItem.Picture

TPictureCollectionItem

```
property Picture: TPicture;
```

Description

It is an image . The content changed until the Restore method is called even if the image is changed is not reflected in the surface.

TPictureCollectionItem.SkipHeight

TPictureCollectionItem

```
property SkipHeight: Integer;
```

Description

This property specifies whether to be spread over the image as the pattern has the space of what pixel in the direction of Y.

TPictureCollectionItem.SkipWidth

TPictureCollectionItem

```
property SkipWidth: Integer;
```

Description

This property specifies whether to be spread over the image as the pattern has the space of what pixel in the direction of X.

TPictureCollectionItem.SystemMemory

TPictureCollectionItem

```
property SystemMemory: Boolean;
```

Description

Whether the surface is secured for the system memory is specified.

TPictureCollectionItem.Transparent

TPictureCollectionItem

```
property Transparent: Boolean;
```

Description

Whether a transparent color is pulled out when describing is specified.

See also

TransparentColor

TPictureCollectionItem.TransparentColor

TPictureCollectionItem

```
property TransparentColor: TColor;
```

Description

It is a transparent color of the image.

See also

Transparent

TPictureCollectionItem.Width

TPictureCollectionItem

```
property Width: Integer;
```

Description

It is actual width of the pattern.

TPictureCollectionItem methods

TPictureCollectionItem   Lengend

In TPictureCollectionItem

- Draw
- DrawAdd
- DrawAlpha
- DrawRotate
- DrawRotateAdd
- DrawRotateAlpha
- DrawRotateSub
- DrawSub
- DrawWaveX
- DrawWaveXAdd
- DrawWaveXAlpha
- DrawWaveXSub
- Restore
- StretchDraw

TPictureCollectionItem.Draw

TPictureCollectionItem

```
procedure Draw(Dest: TDirectDrawSurface; X, Y, PatternIndex: Integer);
```

Description

The pattern is described . To pull out a transparent color,   True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X	X coordinates description ahead
Y	Y coordinates description ahead
PatternIndex	Number of described pattern

TPictureCollectionItem.DrawAdd

TPictureCollectionItem

```
procedure DrawAdd(Dest: TDirectDrawSurface; const DestRect: TRect;
  PatternIndex: Integer; Alpha: Integer=255);
```

Description

The specified pattern is synthesized in addition . However,　the addition synthesis is not done when the number of screen colors is below 256 colors.

To pull out a transparent color,　the Transparent property is set in True.

Argument	Explanation
Dest	Surface description ahead
DestRect	Rectangle description ahead
PatternIndex	Number of described pattern
Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance,　after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified,　the addition synthesis is done.

TPictureCollectionItem.DrawAlpha

TPictureCollectionItem

```
procedure DrawAlpha(Dest: TDirectDrawSurface; const DestRect: TRect;
  PatternIndex: Integer; Alpha: Integer);
```

Description

The alpha-blending does the specified pattern . However, blend is not done when the number of screen colors is below 256 colors.

To pull out a transparent color, the Transparent property is set in True.

Argument	Explanation
Dest	Surface description ahead
DestRect	Rectangle description ahead
PatternIndex	Number of described pattern
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TPictureCollectionItem.DrawRotate

TPictureCollectionItem

```
procedure DrawRotate(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: Integer; CenterX, CenterY: Double; Angle: Integer);
```

Description

The specified pattern is rotating described.

To pull out a transparent color,   True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
PatternIndex	Number of described pattern
Width, Height	Size of destination
CenterX, CenterY	The center of the surface when rotating is specified.
Angle	The rotation angle is specified at 256 cycles.

TPictureCollectionItem.DrawRotateAdd

TPictureCollectionItem

```
procedure DrawRotateAdd(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: Integer; CenterX, CenterY: Double; Angle: Integer; Alpha:
  Integer=255);
```

Description

After the specified pattern is rotated,　the addition synthesis is done . However,　the addition synthesis cannot be done when the number of screen colors is below 256 colors.

To pull out a transparent color,　True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
PatternIndex	Number of described pattern
Width, Height	Size of destination
CenterX, CenterY	The center of the surface when rotating is specified.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance,　after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified,　the addition synthesis is done.

TPictureCollectionItem.DrawRotateAlpha

TPictureCollectionItem

```
procedure DrawRotateAlpha(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: Integer; CenterX, CenterY: Double; Angle, Alpha: Integer);
```

Description

After the specified pattern is rotated, the Transparent a half synthesis is done. However, the addition synthesis cannot be done when the number of screen colors is below 256 colors.

To pull out a transparent color, True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
PatternIndex	Number of described pattern
Width, Height	Size of destination
CenterX, CenterY	The center of the surface when rotating is specified.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TPictureCollectionItem.DrawRotateSub

TPictureCollectionItem

```
procedure DrawRotateSub(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: Integer; CenterX, CenterY: Double; Angle: Integer; Alpha:
  Integer=255);
```

Description

After the specified pattern is rotated,　the subtraction synthesis is done . However,　the subtraction synthesis cannot be done when the number of screen colors is below 256 colors.

To pull out a transparent color,　True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
PatternIndex	Number of described pattern
Width, Height	Size of destination
CenterX, CenterY	The center of the surface when rotating is specified.
Angle	The rotation angle is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance,　after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified,　the subtraction synthesis is done.

TPictureCollectionItem.DrawSub

TPictureCollectionItem

```
procedure DrawSub(Dest: TDirectDrawSurface; const DestRect: TRect;
  PatternIndex: Integer; Alpha: Integer=255);
```

Description

The specified pattern is synthesized in subtraction . However, the subtraction synthesis is not done when the number of screen colors is below 256 colors.

To pull out a transparent color, the Transparent property is set in True.

Argument	Explanation
Dest	Surface description ahead
DestRect	Rectangle description ahead
PatternIndex	Number of described pattern
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the subtraction synthesis is done.

TPictureCollectionItem.DrawWaveX

TPictureCollectionItem

```
procedure DrawWaveX(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: amp, Len, ph: Integer);
```

Description

The shape of waves is transformed into the direction of X and the specified pattern is described.

To pull out a transparent color, True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
Width, Height	Size of destination
PatternIndex	Number of described pattern
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.

TPictureCollectionItem.DrawWaveXAdd

TPictureCollectionItem

```
procedure DrawWaveXAdd(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: amp, Len, ph: Integer; Alpha: Integer=255);
```

Description

The specified pattern is transformed into the direction of X in the shape of waves and the addition synthesis is done . However, the addition synthesis is not done when the number of screen colors is below 256 colors.

To pull out a transparent color, True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
Width, Height	Size of destination
PatternIndex	Number of described pattern
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the addition synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the addition synthesis is done.

TPictureCollectionItem.DrawWaveXAlpha

TPictureCollectionItem

```
procedure DrawWaveXAlpha(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: amp, Len, ph, Alpha: Integer);
```

Description

The specified pattern is done and the shape of waves is transformed into the direction of X and alpha-blending is done . However, blend is not done when the number of screen colors is below 256 colors.

To pull out a transparent color, True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
Width, Height	Size of destination
PatternIndex	Number of described pattern
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.
Alpha	The transparency is specified within the range from 0(transparency) to 255(opacity).

TPictureCollectionItem.DrawWaveXSub

TPictureCollectionItem

```
procedure DrawWaveXSub(Dest: TDirectDrawSurface; X, Y, Width, Height,
  PatternIndex: amp, Len, ph: Integer; Alpha: Integer=255);
```

Description

The specified pattern is transformed into the direction of X in the shape of waves and the subtraction synthesis is done . However,   the subtraction synthesis is not done when the number of screen colors is below 256 colors.

To pull out a transparent color,   True is set in the Transparent property.

Argument	Explanation
Dest	Surface description ahead
X, Y	Coordinates of destination
Width, Height	Size of destination
PatternIndex	Number of described pattern
amp	The maximum value of the amplitude is specified.
Len	The wave length is specified by the number of lines.
ph	The phase is specified at 256 cycles.
Alpha	The ratio of the brightness of the forwarding origin of doing the subtraction synthesis is specified within the range from 0 to 255. For instance, after the pixel in the forwarding origin is made the brightness of 25% when 64 is specified, the subtraction synthesis is done.

TPictureCollectionItem.Restore

TPictureCollectionItem

procedure Restore;

Description

The image is loaded into the surface.

TPictureCollectionItem.StretchDraw

TPictureCollectionItem

```
procedure StretchDraw(Dest: TDirectDrawSurface; const DestRect: TRect;
  PatternIndex: Integer);
```

Description

The pattern reduces expanding and is described.

To pull out a transparent color, the Transparent property is set in True.

Argument	Explanation
Dest	Surface description ahead
DestRect	Rectangle description ahead
PatternIndex	Number of described pattern

DXInput unit

The class concerning the input is declared to the DXInput unit.
The following item is declared to the DXInput unit.

Components

TDXInput

Objects

TCustomInput

TForceFeedbackEffect

TForceFeedbackEffects

TJoystick

TKeyboard

TMouse

TDXInput

Hierarchy Properties Methods Example

Unit

DXInput

Description

The TDXInput component controls two or more input devices . The input device being supported now is a joystick and a keyboard.

When the States property is used, the input situation is obtained . To make the input situation the latest one, the Update method is called.

Hierarchy

TComponent

TCustomDXInput

TDXInput properties

TDXInput Lengend

In TDXInput

ActiveOnly

Joystick

Keyboard

Mouse

States

UseDirectInput

TDXInput.ActiveOnly

TDXInput

```
property ActiveOnly: Boolean;
```

Description

It is specified whether to accept the input only at time when the form where the TDXInput component exists has the foucs.

TDXInput.Joystick

TDXInput

```
property Joystick: TJoystick;
```

Description

It is a joystick.

TDXInput.Keyboard

TDXInput

```
property Keyboard: TKeyboard;
```

Description

It is a keyboard.

TDXInput.Mouse

TDXInput

```
property Mouse: TMouse;
```

Description

It is a Mouse.

TDXInput.States

TDXInput

```
type TDXInputState = set of (isUp, isDown, isLeft, isRight, isButton1,
 isButton2, isButton3, isButton4, isButton5, isButton6, isButton7, isButton8,
 isButton9, isButton10, isButton11, isButton12, isButton13, isButton14,
 isButton15, isButton16, isButton17, isButton18, isButton19, isButton20,
 isButton21, isButton22, isButton23, isButton24, isButton25, isButton26,
 isButton27, isButton28, isButton29, isButton30, isButton31, isButton32);
```

```
property States: TDXInputStates;
```

Description

It is a present input situation . To Update the input situation up to date,   the Update method is called.

Identifier	Meaning
isUp	The up key is pushed.
isDown	The down key is pushed.
isLeft	The left key is pushed.
isRight	The right key is pushed.
isButton1	Button 1 is pushed.
isButton32	Button 32 is pushed.

TDXInput.UseDirectInput

TDXInput

```
property UseDirectInput: Boolean;
```

Description

It is specified whether to use it automatically if DirectInput exists in the execution environment.
It is necessary to set this property in True to use the force feedback etc.

TDXInput methods

TDXInput Lenqend

In TDXInput

Update

TDXInput.Update

TDXInput

```
procedure Update;
```

Description

A present input situation is read from two or more input devices, and the one that they were synthesized to the States property is stored.

Example of TDXInput

The label is moved . And, when button 1 is pushed, the alarm is sounded.

```
procedure TForm1.Timer1Event(Sender: TObject);
begin
  DXInput1.Update;

  if isLeft in DXInput1.States then
    Label1.Left := Label1.Left - 10;

  if isRight in DXInput1.States then
    Label1.Left := Label1.Left + 10;

  if isUp in DXInput1.States then
    Label1.Top := Label1.Top - 10;

  if isDown in DXInput1.States then
    Label1.Top := Label1.Top + 10;

  if isButton1 in DXInput1.States then
  begin
    Beep;

    {  Next,  button 1 is invalidated until button 1 is pushed.  }
    DXInput1.States := DXInput1.States - [isButton1];
  end;
end;
```

TCustomInput

Hierarchy Properties Methods

Unit

DXInput

Description

The TCustomInput object is an abstraction class of the input device . The TJoystick object and the TKeyboard object have succeeded to TCustomInput.

The ButtonCount property is used for obtaining the number of buttons of this device . The Buttons property is used to acquire which button is being pushed now.

To make the force feedback function of this device effective, the ForceFeedback property is set in True . Effect of the force feedback can be set with the Effects property.

Hierarchy

TPersistent

TCustomInput properties

TCustomInput Lenqend

In TCustomInput

BindInputStates

ButtonCount

Buttons

Effects

ForceFeedback

States

TCustomInput.BindInputStates

TCustomInput

```
property BindInputStates: Boolean;
```

Description

It is specified whether to reflect the value of the States property in the States property of the TDXInput component.

TCustomInput.ButtonCount

TCustomInput

```
property ButtonCount: Integer;
```

Description

It is a number of buttons of this input device.

TCustomInput.Buttons

TCustomInput

```
property Buttons[Index: Integer]: Boolean;
```

Description

It is returned whether the button specified now is pushed.

TCustomInput.Effects

TCustomInput

```
property Effects: TForceFeedbackEffects;
```

Description

It is Effect of the force feedback.

TCustomInput.ForceFeedback

TCustomInput

```
property ForceFeedback: Boolean;
```

Description

It is specified whether to use the force feedback.

See also

TDXInput.UseDirectInput

TCustomInput.States

TCustomInput

```
property States: TDXInputStates;
```

Description

It is a present input situation . To update the input situation to the latest one,   the Update method is called.

TCustomInput methods

TCustomInput Lengend

In TCustomInput

Update

TCustomInput.Update

TCustomInput

```
procedure Update;
```

Description

The input situation is read from the input device.

TForceFeedbackEffect

Hierarchy Properties Methods

Unit

DXInput

Description

It is Effect of the force feedback.

The EffectType property is used to specify the kind of executed effect.

To call the Start method to execute effect and to stop, the Stop method is called.

Hierarchy

TPersistent

TForceFeedbackEffect properties

TForceFeedbackEffect Lenqend

In TForceFeedbackEffect

AttackLevel

AttackTime

Condition

Constant

Count

Effects

EffectType

FadeLevel

FadeTime

Name

Parent

Period

Power

Time

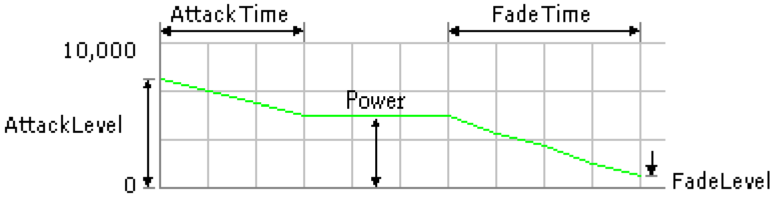
TForceFeedbackEffect.AttackLevel

TForceFeedbackEffect

```
property AttackLevel: Integer;
```

Description

Strength when Effect starts is specified within the range of 0-10000.



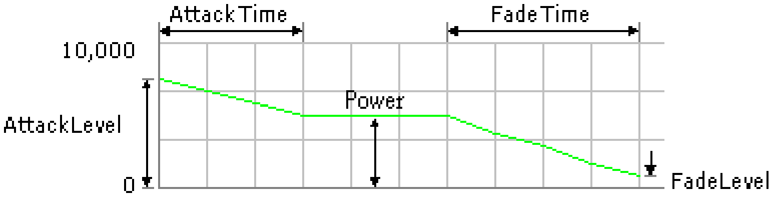
TForceFeedbackEffect.AttackTime

TForceFeedbackEffect

```
property AttackTime: Integer;
```

Description

The strength transition time when Effect starts is specified in each millisecond.



TForceFeedbackEffect.Condition

TForceFeedbackEffect

```
property Condition: TPoint;
```

Description

When the EffectType property is etCondition,   this property is effective .
How to put the power when the stick is centered is specified within the range of -10000 to +10000.

TForceFeedbackEffect.Constant

TForceFeedbackEffect

```
property Constant: TPoint;
```

Description

The constant of Effect . The meaning changes depending on the value of the EffectType property.

Examples

For etConstantForce

How to put power . For instance, when Point(0, -10000) is specified, power for above is put on the stick by the maximum power.

For etPeriodic

How to put power and direction of vibration . For instance, when Point(0, -10000) is specified, the stick is vibrated along Y axis by the maximum power.

For etCondition

The starting point when the stick is centered . For instance, when Point(0, -10000) is specified, the left side is centered as a starting point.

TForceFeedbackEffect.Count

TForceFeedbackEffect

```
property Count: Integer;
```

Description

The number of maintained Effect is returned.

TForceFeedbackEffect.Effects

TForceFeedbackEffect

```
property Effects[Index: Integer]: TForceFeedbackEffect; default;
```

Description

It is a list of Effect being maintained now.

TForceFeedbackEffect.EffectType

TForceFeedbackEffect

```
TForceFeedbackEffectType = set of (etNone, etConstantForce, etRampForce,
  etPeriodic, etCondition);
```

```
property EffectType: TForceFeedbackEffectType;
```

Description

The kind of Effect is specified.

Identifier	Meaning
etNone	nothing is done.
etConstantForce	power is put. -> Constant
etPeriodic	vibrates. -> Constant, Period
etCondition	centering and weight are controlled. > Condition, Constant

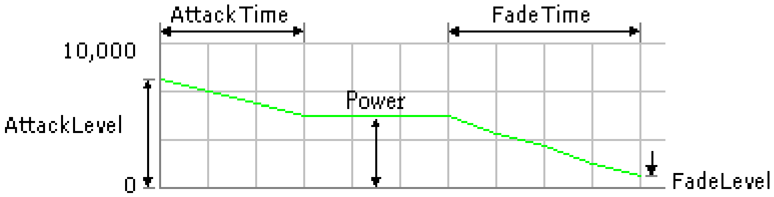
TForceFeedbackEffect.FadeLevel

TForceFeedbackEffect

```
property FadeLevel: Integer;
```

Description

Strength when Effect is ended is specified within the range of 0-10000.



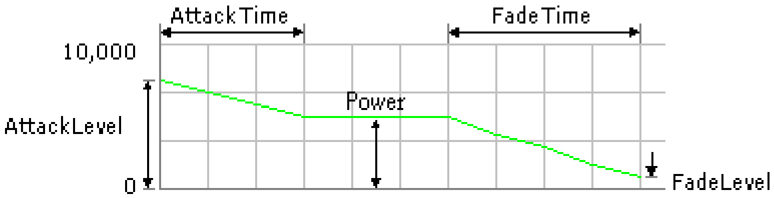
TForceFeedbackEffect.FadeTime

TForceFeedbackEffect

```
property FadeTime: Integer;
```

Description

The strength transition time when Effect is ended is specified in each millisecond.



TForceFeedbackEffect.Name

TForceFeedbackEffect

```
property Name: string;
```

Description

It is a name of this Effect.

TForceFeedbackEffect.Parent

TForceFeedbackEffect

```
property Parent: TForceFeedbackEffect;
```

Description

It is a parent of this Effect.

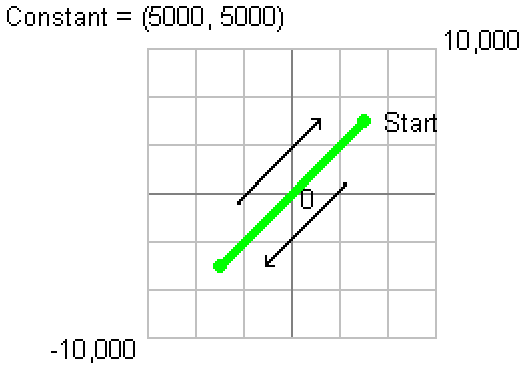
TForceFeedbackEffect.Period

TForceFeedbackEffect

```
property Period: Integer;
```

Description

When the EffectType property is etPeriodic,   this property is effective .
The speed at the vibration is specified in each millisecond.



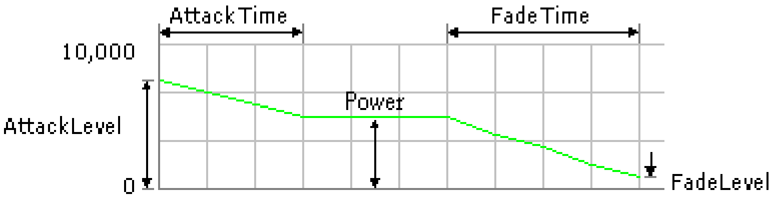
TForceFeedbackEffect.Power

TForceFeedbackEffect

```
property Power: Integer;
```

Description

Strength of Effect is specified within the range of 0-10000. Effect strengthens by there are a lot of values.



TForceFeedbackEffect.Time

TForceFeedbackEffect

```
property Time: Integer;
```

Description

Length by which Effect is executed is specified in each millisecond . When -1 is specified,   Effect is infinitely executed.

TForceFeedbackEffect methods

TForceFeedbackEffect Lengend

In TForceFeedbackEffect

Clear

Create

Destroy

Find

IndexOf

Start

Stop

TForceFeedbackEffect.Clear

TForceFeedbackEffect

```
procedure Clear;
```

Description

All Effect being maintained now is abandoned.

TForceFeedbackEffect.Create

TForceFeedbackEffect

```
constructor Create(AParent: TForceFeedbackEffect);
```

Description

The TForceFeedbackEffect object is made.

Argument	Explanation
AParent	Parent

TForceFeedbackEffect.Destroy

TForceFeedbackEffect

```
destructor Destroy;
```

Description

The TForceFeedbackEffect object is abandoned.

TForceFeedbackEffect.Find

TForceFeedbackEffect

```
procedure Find(const Name: string): TForceFeedbackEffect;
```

Description

Effect of the specified name is returned . Effect of a deep hierarchy can be acquired by delimiting the name at the period.

TForceFeedbackEffect.IndexOf

TForceFeedbackEffect

```
procedure IndexOf(const Name: string): Integer;
```

Description

The index of Effect of the specified name is returned.

TForceFeedbackEffect.Start

TForceFeedbackEffect

```
procedure Start;
```

Description

All Effect maintained now is started . When the Time property is -1,   Effect is not stopped until the Stop method is called.

TForceFeedbackEffect.Stop

TForceFeedbackEffect

```
procedure Stop;
```

Description

All Effect maintained now is stopped.

TForceFeedbackEffects

Hierarchy Properties Methods

Unit

DXInput

Description

The TForceFeedbackEffects object controls the TForceFeedbackEffect object.

Hierarchy

TPersistent

TForceFeedbackEffect

TForceFeedbackEffects properties

TForceFeedbackEffects   Lenqend

Derived from TForceFeedbackEffect

AttackLevel

AttackTime

Condition

Constant

Count

Effects

EffectType

FadeLevel

FadeTime

Name

Parent

Period

Power

Time

TForceFeedbackEffects methods

TForceFeedbackEffects Lengend

Derived from TForceFeedbackEffect

Clear

Destroy

Find

IndexOf

Start

Stop

TJoystick

Hierarchy Properties Methods

Unit

DXInput

Description

The TKeyboard object controls the joystick.

It can be specified whether to use the device connected with how many in the ID property.

The coordinates value of the axis of a present stick can be acquired in X, Y, Z, and the Joystate property. RangeX, RangeY, RangeZ, and the Range property are used to specify the range of the value.

The ButtonCount property is used for obtaining the number of buttons of this device . The Buttons property is used to acquire which button is being pushed now.

To make the force feedback function of this device effective, the ForceFeedback property is set in True . Effect of the force feedback can be set with the Effects property.

Hierarchy

TPersistent

TCustomInput

TJoystick properties

TJoystick Lenqend

In TJoystick

AutoCenter

DeadZone

DeadZoneX

DeadZoneY

DeadZoneZ

ID

Joystate

Range

RangeX

RangeY

RangeZ

X

Y

Z

Derived from TCustomInput

BindInputStates

ButtonCount

Buttons

Effects

ForceFeedback

States

TJoystick.AutoCenter

TJoystick

```
property AutoCenter: Boolean;
```

Description

It is specified whether to center the stick effectively.

TJoystick.DeadZone

TJoystick

```
property DeadZone[Obj: Integer]: Integer;
```

Description

How many percent of the sensitivity of the axis specified by the Obj argument is dropped is specified . For instance,　if 80 is specified,　sensitivity considerably becomes a hebetude.

When the specified object does not exist or DeadZone cannot be set in the acquisition of the value of the DeadZone property,　-1 is returned.

The offset value of the field of DIJOYSTATE2 structural body is specified for the Obj argument . For instance,　to set the range of the value of the Joystate.lRx field,　as follows is done.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  DXInput1.DeadZone[Integer(@PDIJOYSTATE2(nil).lRx)] := 80;
end;
```

TJoystick.DeadZoneX

TJoystick

```
property DeadZoneX: Integer;
```

Description

How many percent of the sensitivity of X axis is dropped is specified . For instance, if 80 is specified, sensitivity considerably becomes a hebetude.

TJoystick.DeadZoneY

TJoystick

```
property DeadZoneY: Integer;
```

Description

How many percent of the sensitivity of Y axis is dropped is specified . For instance,   if 80 is specified, sensitivity considerably becomes a hebetude.

TJoystick.DeadZoneZ

TJoystick

```
property DeadZoneZ: Integer;
```

Description

How many percent of the sensitivity of Z axis is dropped is specified . For instance, if 80 is specified, sensitivity considerably becomes a hebetude.

TJoystick.ID

TJoystick

```
property ID: Integer;
```

Description

It is ID of the joystick . One public notice is 0 Two public notices become one.

TJoystick.Joystate

TJoystick

```
property Joystate: DIJOYSTATE2;
```

Description

It is a state of the joystick . DIJOYSTATE2 structural body is declared for the DirectX unit.

TJoystick.Range

TJoystick

```
property Range[Obj: Integer]: Integer;
```

Description

The range of the value of the axis specified by the Obj argument is specified. For instance, the range of the axis coordinates value specified that 1000 is specified becomes 1000 from -1000.

When the specified object does not exist or the range cannot be set in the acquisition of the value of the Range property, -1 is returned.

The offset value of the field of DIJOYSTATE2 structural body is specified for the Obj argument. For instance, to set the range of the value of the Joystate.lRx field, as follows is done.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  DXInput1.Range[Integer(@PDIJOYSTATE2(nil).lRx)] := 1000;
end;
```

TJoystick.RangeX

TJoystick

```
property RangeX: Integer;
```

Description

The range of X axis is specified . For instance, if 1000 is specified, the value of X property becomes within the range of -1000 to +1000.

TJoystick.RangeY

TJoystick

```
property RangeY: Integer;
```

Description

The range of Y axis is specified . For instance, if 1000 is specified, the value of Y property becomes within the range of -1000 to +1000.

TJoystick.RangeZ

TJoystick

```
property RangeZ: Integer;
```

Description

The range of Z axis is specified . For instance,　if 1000 is specified,　the value of Z property becomes within the range of -1000 to +1000.

TJoystick.X

TJoystick

```
property Z: Integer;
```

Description

It is coordinates of Z axis of the joystick.

TJoystick.Y

TJoystick

```
property Y: Integer;
```

Description

It is coordinates of Y axis of the joystick.

TJoystick.Z

TJoystick

```
property Z: Integer;
```

Description

It is coordinates of Z axis of the joystick.

TJoystick methods

TJoystick Lenqend

Derived from TCustomInput

Update

TKeyboard

Hierarchy Properties Methods

Unit

DXInput

Description

The TKeyboard object controls the keyboard.

The Keys property is used to acquire which key is being pushed now.

Which key is allocated input when the KeyAssigns property is used can be specified.

To make the force feedback function of this device effective,　the ForceFeedback property is set in True . Effect of the force feedback can be set with the Effects property.

Hierarchy

TPersistent

TCustomInput

TKeyboard properties

TKeyboard Legend

In TKeyboard

KeyAssigns

Keys

Derived from TCustomInput

BindInputStates

ButtonCount

Buttons

Effects

ForceFeedback

States

TKeyboard.KeyAssigns

TKeyboard

```
property KeyAssigns: array[TDXInputStates, 0..2] of Integer;
```

Description

It is an allocation of the key list . Keys are allocated to one input up to three.

TKeyboard.Keys

TKeyboard

```
property Keys[Index: Integer]: Boolean;
```

Description

It is returned whether the key specified now is pushed.

TKeyboard methods

TKeyboard Lenqend

Derived from TCustomInput

Update

TMouse

Hierarchy Properties Methods

Unit

DXInput

Description

The TKeyboard object controls the mouse.

The States property is used to acquire which button is being pushed now.

The amount of the movement of the mouse can be acquired in X property, Y property, and Z property.

Hierarchy

TPersistent

TCustomInput

TMouse properties

TMouse Lengend

In TMouse

X

Y

Z

Derived from TCustomInput

BindInputStates

ButtonCount

Buttons

Effects

ForceFeedback

States

TMouse.X

TMouse

```
property X: Integer;
```

Description

It is an amount of the movement of X axis after the Update method of last time is called.

TMouse.Y

TMouse

```
property Y: Integer;
```

Description

It is an amount of the movement of Y axis after the Update method of last time is called.

TMouse.Z

TMouse

```
property Z: Integer;
```

Description

It is an amount of the movement of Z axis after the Update method of last time is called.

TMouse methods

TMouse Lenqend

Derived from TCustomInput

Update

DXPlay unit

The class concerning the communication is declared to the DXInput unit.
The following item is declared to the DXInput unit.

Components

TDXPlay

Objects

TDXPlayModemSetting

TDXPlayPlayer

TDXPlayPlayers

TDXPlayTCPIPSetting

TDXPlay

Hierarchy Properties Methods Events

Unit

DXPlay

Description

The TDXPlay component is a component to communicate.

Peculiar GUID to the application can be specified with the GUID property.

The communication of the Open method and the Open2 method begins . Moreover, the communication can be interrupted by the Close method.

The message can be sent to other players by using the SendMessage method. Moreover, when the message is received, the OnMessage event is generated.

Hierarchy

TComponent

TCustomDXPlay

TDXPlay properties

TDXPlay Lenqend

In TDXPlay

GUID

IsHost

LocalPlayer

MaxPlayers

ModemSetting

Opened

Players

ProviderName

Providers

Sessions

TCPIPSetting

TDXPlay.GUID

TDXPlay

```
property GUID: string;
```

Description

It is GUID of the application . The character string like {00000000-0000-0000-0000-000000000000} is specified.

Please specify unique GUID in all over the world.

TDXPlay.IsHost

TDXPlay

```
property IsHost: Boolean;
```

Description

It is returned whether it is a present host.

TDXPlay.LocalPlayer

TDXPlay

```
property LocalPlayer: TDXPlayPlayer;
```

Description

It is a player of the local.

TDXPlay.MaxPlayers

TDXPlay

```
property MaxPlayers: Integer;
```

Description

It is the maximum number of players which can participate . There is no limitation in the number of players when 0 is specified.

TDXPlay.ModemSetting

TDXPlay

```
property ModemSetting: TDXPlayModemSetting;
```

Description

It is setting of the modem.

If this property is effectively set when the modem is specified for provider by the GetSessions method and the Open2 method, the diamond log is not displayed.

TDXPlay.Opened

TDXPlay

```
property Opened: Boolean;
```

Description

It is returned whether to be being opened now.

TDXPlay.Players

TDXPlay

```
property Players: TDXPlayPlayers;
```

Description

It is a list of the player which is participating now.

TDXPlay.ProviderName

TDXPlay

```
property ProviderName: string;
```

Description

It is a name of the provider being connected now.

When this property is set, the TDXPlay component initializes DirectPlay . At this time, the content of the XXXSetting property is used.

Using GetSessions and the Open2 method only after this property is set becomes possible.

TDXPlay.Providers

TDXPlay

```
property Providers: TStrings;
```

Description

It is a list of provider which can be used now.

TDXPlay.Sessions

TDXPlay

```
property Sessions: TStrings;
```

Description

It is a list of the session being connected now.
To update this list to the latest one,　the GetSessions method is called.

TDXPlay.TCPIPSetting

TDXPlay

```
property TCPIPSetting: TDXPlayTCPIPSetting;
```

Description

It is setting of TCP/IP.

If this property is effectively set when TCP/IP is specified for provider by the GetSessions method and the Open2 method,　the diamond log is not displayed.

TDXPlay methods

TDXPlay Lengend

In TDXPlay

Close

GetSessions

Open

Open2

SendMessage

TDXPlay.Close

TDXPlay

```
procedure Close;
```

Description

The communication is interrupted . At this time, the OnClose event is generated.

TDXPlay.GetSessions

TDXPlay

```
procedure GetSessions;
```

Description

The content of the Sessions property is updated to the latest one . When this method is called,   it is necessary to set the ProviderName property.

TDXPlay.Open

TDXPlay

```
procedure Open;
```

Description

The communication begins . At this time, the OnOpen event is generated.
This method disregards the ProviderName property.

TDXPlay.Open2

TDXPlay

```
procedure Open2(const NewSession: Boolean; const SessionName, PlayerName:
  string);
```

Description

The communication begins according to the argument specified without displaying the diamond log like the Open method . When this method is called,　it is necessary to set the ProviderName property.

At this time,　the content of the XXXSetting property is used.

Argument	Explanation
NewSession	Whether the session is newly made or not?
SessionName	When True is specified for the NewSession argument,　the SessionName argument becomes a session name newly made . When False is specified oppositely,　the SessionName argument becomes a session name the connection ahead . The list of the session is obtained by the GetSessions method.
PlayerName	Player name

TDXPlay.SendMessage

TDXPlay

```
procedure SendMessage(ToID: DPID; Data: Pointer; DataSize: Integer);
```

Description

The message is sent . When DPID_ALLPLAYERS is specified for the ToID argument, the message is sent to all players except me.

Argument	Explanation
ToID	ID of address of sent message
Data	Pointer to sent message
DataSize	Size of sent message

TDXPlay events

TDXPlay Lengend

In TDXPlay

OnAddPlayer

OnClose

OnDeletePlayer

OnMessage

OnOpen

OnSessionLost

TDXPlay.OnAddPlayer

TDXPlay

```
TDXPlayEvent = procedure(Sender: TObject; Player: TDXPlayPlayer) of object;
property OnAddPlayer: TDXPlayEvent;
```

Description

When other players have participated newly,   this event is generated.

Argument	Explanation
Player	Player which has participated newly

TDXPlay.OnClose

TDXPlay

```
property OnClose: TNotifyEvent;
```

Description

When the communication is interrupted,   this event is generated.

TDXPlay.OnDeletePlayer

TDXPlay

```
TDXPlayEvent = procedure(Sender: TObject; Player: TDXPlayPlayer) of object;
property OnDeletePlayer: TDXPlayEvent;
```

Description

When other players come off the game,　this event is generated.

Argument	Explanation
Player	Player which came off game

TDXPlay.OnMessage

TDXPlay

```
TDXPlayMessageEvent = procedure(Sender: TObject; From: TDXPlayPlayer; Data:
  Pointer; DataSize: Integer) of object;
```

```
property OnMessage: TDXPlayMessageEvent;
```

Description

When the message reaches,   this event is generated.

Argument	Explanation
Form	Player by which message is given
Data	Pointer to message
DataSize	Size of message

TDXPlay.OnOpen

TDXPlay

```
property OnOpen: TNotifyEvent;
```

Description

When the communication begins,　this event is generated.

TDXPlay.OnSessionLost

TDXPlay

```
property OnSessionLost: TNotifyEvent;
```

Description

When the communication is interrupted from the outside,   this event is generated.

TDXPlayModemSetting

Hierarchy Properties

Unit

DXPlay

Description

When the modem is used,　the TDXPlayModemSetting object maintains necessary information.

The name of the modem used with the ModemName property can be specified . The list of the modem which can be used now is obtained by the ModemNames property.

The telephone number the connection ahead can be specified with the PhoneNumber property.

Hierarchy

TPersistent

TDXPlayModemSetting properties

TDXPlayModemSetting Lengend

In TDXPlayModemSetting

Enabled

ModemName

ModemNames

PhoneNumber

TDXPlayModemSetting.Enabled

TDXPlayModemSetting

```
property Enabled: Boolean;
```

Description

It is specified whether to use setting this modem.

DirectPlay should display the diamond log when False is specified for this property, and the user set information necessary to use the modem.

TDXPlayModemSetting.ModemName

TDXPlayModemSetting

```
property ModemName: string;
```

Description

It is a name of the modem used . The list of the modem which can be used now is obtained by the ModemNames property.

TDXPlayModemSetting.ModemNames

TDXPlayModemSetting

```
property ModemNames: TStrings;
```

Description

It is a list of the modem which can be used now.

TDXPlayModemSetting.PhoneNumber

TDXPlayModemSetting

```
property PhoneNumber: string;
```

Description

It is a telephone number the connection ahead.

TDXPlayPlayer

Hierarchy Properties

Unit

DXPlay

Description

The TDXPlayPlayer object shows the player of the TDXPlay component.
The Name property is used to acquire the name of this player.
The Data property is used to relate the data of the application definition to the player.

Hierarchy

TCollectionItem

TDXPlayPlayer properties

TDXPlayPlayer   Lenqend

In TDXPlayPlayer

- Data
- ID
- Name
- RemotePlayer

TDXPlayPlayer.Data

TDXPlayPlayer

```
property Data: Pointer;
```

Description

The Data property is a pointer to the data of the application definition.

The data of the application definition can be related to the player by using this property.

TDXPlayPlayer.ID

TDXPlayPlayer

```
property ID: DPID;
```

Description

It is ID of the player . The value of this property can be used by the SendMessage method of the TDXPlay component.

TDXPlayPlayer.Name

TDXPlayPlayer

```
property Name: string;
```

Description

It is a name of this player.

TDXPlayPlayer.RemotePlayer

TDXPlayPlayer

```
property RemotePlayer: Boolean;
```

Description

It is returned whether this player is a remote player.

TDXPlayPlayers

Hierarchy   Properties

Unit

DXPlay

Description

The TDXPlayPlayers object is a list of the TDXPlayPlayer object.

Hierarchy

TCollection

TDXPlayPlayers properties

TDXPlayPlayers   Lenqend

In TDXPlayPlayers

Count

Players

TDXPlayPlayers.Count

TDXPlayPlayers

```
property Count: Integer;
```

Description

A present number of players is returned.

TDXPlayPlayers.Players

TDXPlayPlayers

```
property Players[Index: Integer]: TDXPlayPlayer; default;
```

Description

It is a list of the player.

TDXPlayTCPIPSetting

Hierarchy Properties

Unit

DXPlay

Description

When TCP/IP is used,　the TDXPlayTCPIPSetting object maintains necessary information.
The address connecting ahead can be specified with the HostName property.

Hierarchy

TPersistent

TDXPlayTCPIPSetting properties

TDXPlayTCPIPSetting Lengend

In TDXPlayTCPIPSetting

Enabled

HostName

TDXPlayTCPIPSetting.Enabled

TDXPlayTCPIPSetting

```
property Enabled: Boolean;
```

Description

It is specified whether to use setting this TCP/IP.

DirectPlay should display the diamond log when False is specified for this property,   and the user set information necessary to use TCP/IP.

TDXPlayTCPIPSetting.HostName

TDXPlayTCPIPSetting

```
property HostName: string;
```

Description

The address connecting ahead is specified by the dot form (196.168.1.1 etc.) or the name.

DXSounds unit

The class concerning DirectSound is declared to the DXSounds unit.
The following item is declared to the DXSounds unit.

Components

TDXSound

TDXWaveList

Objects

TAudioFileStream

TAudioStream

TDirectSound

TDirectSoundBuffer

TSoundCaptureFormat

TSoundCaptureFormats

TSoundCaptureStream

TWaveCollection

TWaveCollectionItem

TDXSound

Hierarchy Properties Methods Events

Unit

DXSounds

Description

The TDXSound component can easily use DirectSound.

When not muting when the form where the TDXSound component exists loses the foucs, doGlobalFocus of the Options property or doStickyFocus is made True.

The reproduction sound quality of default is 22050Hz 8 bit stereo . Please refer to The reproduction quality of TDXSound is set. to change the quality.

Hierarchy

TComponent

TCustomDXSound

TDXSound properties

TDXSound Lenqend

In TDXSound

AutoInitialize

Driver

DSound

Initialized

NowOptions

Options

Primary

TDXSound.AutoInitialize

TDXSound

```
property AutoInitialize: Boolean;
```

Description

It is specified whether to call the Initialize method automatically when the application is started . TDXSound conceals it even if the exception is generated at that time.

Please call the Initialize method without using the AutoInitialize property yourself so as not to conceal the exception because of a form OnCreate event.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
  DXSound1.Initialize;
end;
```

TDXSound.Driver

TDXSound

```
property Driver: PGUID;
```

Description

It is a driver of DirectSound used when the Initialize method is called . When nil is specified,   the driver of default is used.

See also

Drivers

TDXSound.DSound

TDXSound

```
property DSound: TDirectSound;
```

Description

It is TDirectSound object.

TDXSound.Initialized

TDXSound

```
property Initialized: Boolean;
```

Description

It is returned whether DirectSound is being initialized now.

TDXSound.NowOptions

TDXSound

```
property NowOptions: TDXSoundOptions;
```

Description

It is setting of present . The Options property is used to change setting.

TDXSound.Options

TDXSound

```
type TDXSoundOptions = set of (soGlobalFocus, soStickyFocus, soExclusive,
  soWritePrimary);
```

```
property Options: TDXSoundOptions;
```

Description

How of the operation of the TDXSound component can be set by using the Options property . The content changed to the NowOptions property according to the set content is reflected.

The combination of the following values can be specified for the Options property.

Identifier	Meaning
soGlobalFocus	Even if the foucs is lost, the application does not mute the sound . However, when the foucs is switched to the application to which an exclusive level is set at the cooperation level of other DirectSound, the sound is muted.
soStickyFocus	Even if the foucs is lost, the application does not mute the sound . However, when the foucs is switched to other DirectSound applications, the sound is muted.
soExclusive	The application sets an exclusive level at the cooperation level . As a result, the call of the SetFormat method to a primary buffer becomes possible.
soWritePrimary	The application sets an exclusive level of most significant at the cooperation level . Writing a primary buffer in addition to the case to specify soExclusive becomes possible.

TDXSound.Primary

TDXSound

```
property Primary: TDirectSoundBuffer;
```

Description

It is a primary buffer.

TDXSound methods

TDXSound Lenqend

In TDXSound

Drivers

Finalize

Initialize

Restore

TDXSound.Drivers

TDXSound Example

```
class function Drivers: TDirectXDrivers;
```

Description

The list of the DirectSound driver is returned.

See also

Driver

Example of TDXSound.Drivers

DirectSound is initialized by using the driver enumerated first.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  WaveFormat: TWaveFormatEx;
begin
  DXSound1.Driver := TDXSound.Drivers[0].GUID;
  DXSound1.Initialize;
end;
```

TDXSound.Finalize

TDXSound

```
procedure Finalize;
```

Description

The termination of DirectSound is done . At this time, the OnFinalize event is generated.

TDXSound.Initialize

TDXSound

```
function Initialize: Boolean;
```

Description

DirectSound is initialized . At this time,   the event is generated in order of OnInitializing-OnInitialize-OnRestore.

TDXSound.Restore

TDXSound

```
procedure Restore;
```

Description

DirectSound is restored . At this time,   the OnRestore event is generated.

TDXSound events

TDXSound Lengend

In TDXSound

OnFinalize

OnInitialize

OnInitializing

OnRestore

TDXSound.OnFinalize

TDXSound

```
property OnFinalize: TNotifyEvent;
```

Description

It is an event generated when the termination is done . This event has paired with the OnInitialize event .

TDXSound.OnInitialize

TDXSound Example

```
property OnInitialize: TNotifyEvent;
```

Description

It is an event generated when the TDXSound component is initialized . This event has paired with the OnFinalize event .

Please make the buffer here.

TDXSound.OnInitializing

TDXSound

```
property OnInitializing: TNotifyEvent;
```

Description

It is an event generated when the TDXSound component tried to be initialized.

TDXSound.OnRestore

TDXSound

```
property OnRestore: TNotifyEvent;
```

Description

It is an event generated when the content of the sound buffer is lost,   and the necessity for the reload of Wave is caused.

TDXWaveList

Hierarchy   Properties

Unit

DXSounds

Description

The TDXWaveList component is a list of the TWave object . The buffer is controlled by specifying the DXSound property.

Hierarchy

TComponent

TCustomDXWaveList

TDXWaveList properties

TDXWaveList Lenqend

In TDXWaveList

DXSound

Items

TDXWaveList.DXSound

TDXWaveList

```
property DXSound: TDXSound;
```

Description

The TDXSound component is specified . When this property is specified,   the buffer is controlled.

TDXWaveList.Items

TDXWaveList

```
property Items: TWaveCollection;
```

Description

It is a list of the TWave object.

TAudioFileStream

Hierarchy Properties Methods

Unit

DXSounds

Description

The TAudioFileStream object reproduces the Wave file with DirectSound in streamming. Huge Wave can be reproduced by using this object.

The TAudioStream object has achieved the reproduction for a long time by securing seconds of the buffer how many, and pouring the Wave data continuously there. The BufferLength property specifies the length of the buffer.

The Wave file which reproduces with the FileName property can be specified.

The Play method is called so that the reproduction may begin. In that case, to loop, the Looped property is set in True.

To stop reproducing, the Stop method is called.

Hierarchy

TObject

TAudioStream

TAudioFileStream properties

TAudioFileStream Lengend

In TAudioFileStream

FileName

Derived from TAudioStream

AutoUpdate

BufferLength

Format

FormatSize

Frequency

Looped

Pan

Playing

Position

Size

Volume

TAudioFileStream.FileName

TAudioFileStream

```
property FileName: string;
```

Description

The reproducing Wave file name is specified.

TAudioFileStream methods

TAudioFileStream Lengend

In TAudioFileStream

Create

Derived from TAudioStream

Destroy

Play

RecreateBuf

Stop

Update

TAudioFileStream.Create

TAudioFileStream

```
constructor Create(ADSound: TDirectSound);
```

Description

The TAudioFileStream object is made.

Afterwards,　please set the BufferLength property and the FileName property.

Argument	Explanation
ADSound	The TDirectSound object . The value of the TDXSound.DSound property is specified.

TAudioStream

Hierarchy Properties Methods

Unit

DXSounds

Description

The TAudioFileStream object reproduces the Wave stream with DirectSound in streamming. Huge Wave can be reproduced by using this object.

The TAudioStream object has achieved the reproduction for a long time by securing seconds of the buffer how many,　and pouring the Wave data continuously there.　The BufferLength property specifies the length of the buffer.

The Wave stream which reproduces with the WaveStream property can be specified.

The Play method is called so that the reproduction may begin.　In that case,　to loop,　the Looped property is set in True.

To stop reproducing,　the Stop method is called.

Hierarchy

TObject

TAudioStream properties

TAudioStream Lengend

In TAudioStream

AutoUpdate

BufferLength

Format

FormatSize

Frequency

Looped

Pan

Playing

Position

Size

Volume

WaveStream

TAudioStream.AutoUpdate

TAudioStream

```
property AutoUpdate: Boolean;
```

Description

It is specified whether to keep automatically reading the Wave data to the transmission buffer . It is default and True.

When True is specified for the AutoUpdate property, the thread calls the Update method by thread's being started internally every about 50 milliseconds.

When False is specified for the AutoUpdate property, it is necessary to call the Update method in the length or less specified with the BufferLength property for myself periodically.

TAudioStream.BufferLength

TAudioStream

```
property BufferLength: Integer;
```

Description

The size of the transmission buffer is specified in each millisecond . 1000 is specified by default.

The TAudioStream object has achieved the reproduction for a long time by securing seconds of the buffer how many,   and pouring the Wave data in it continuously . This property specifies the length of the buffer.

The sound becomes specification of a big number for this property and the road comes not to cut easily.

TAudioStream.Format

TAudioStream

```
property Format: PWaveFormatEx;
```

Description

When the Format property is used,   information on Wave is obtained . The Format property is secured by the size returned by the FormatSize property.

TAudioStream.FormatSize

TAudioStream

```
property FormatSize: Integer;
```

Description

It is a size for which the Format property is secured.

TAudioStream.Frequency

TAudioStream

```
property Frequency: Integer;
```

Description

It is a reproduction frequency.

TAudioStream.Looped

TAudioStream

```
property Looped: Boolean;
```

Description

It is specified whether to do the loop reproduction.

TAudioStream.Pan

TAudioStream

```
property Pan: Integer;
```

Description

A relative volume of a right and left channel is specified within the range from -10000 to 10000.

A right channel is muted when -10000 is specified,　and when 10000 is specified,　a left channel is muted.

TAudioStream.Playing

TAudioStream

```
property Playing: Boolean;
```

Description

This property is returned whether reproducing now.

TAudioStream.Position

TAudioStream

```
property Position: Integer;
```

Description

The position which has reproduced now is specified by each byte.

TAudioStream.Size

TAudioStream

```
property Size: Integer;
```

Description

The size of Wave is returned . Wave is infinity at -1.

TAudioStream.Volume

TAudioStream

```
property Volume: Integer;
```

Description

The volume is specified within the range from -10000 to 0. It is muted that -10000 is specified,　and if 0 is specified,　becomes a usual volume.

TAudioStream.WaveStream

TAudioStream

```
property WaveStream: TCustomWaveStream;
```

Description

It is a reproducing Wave stream . When this property is set, the Frequency property, the Pan property, and the Volume property are set in the value of default.

TAudioStream methods

TAudioStream Lenqend

In TAudioStream

Create

Destroy

Play

RecreateBuf

Stop

Update

TAudioStream.Create

TAudioStream

```
constructor Create(ADSound: TDirectSound);
```

Description

The TAudioStream object is made.

Afterwards, please set the BufferLength property and the WaveStream property.

Argument	Explanation
ADSound	The TDirectSound object . The value of the TDXSound.DSound property is specified.

TAudioStream.Destroy

TAudioStream

```
destructor Destroy;
```

Description

The TAudioStream object is abandoned.

TAudioStream.Play

TAudioStream

```
procedure Play;
```

Description

The reproduction of the Wave stream begins.

TAudioStream.RecreateBuf

TAudioStream

```
procedure RecreateBuf;
```

Description

The RecreateBuf method is about to make the buffer.

TAudioStream.Stop

TAudioStream

```
procedure Stop;
```

Description

The reproduction of the Wave stream is stopped.

TAudioStream.Update

TAudioStream

```
procedure Update;
```

Description

New Wave data is read to the transmission buffer . In should the call of this method,   the value of the AutoUpdate property is only False.

TDirectSound

Hierarchy Properties Methods

Unit

DXSounds

Description

The wrapping does the IDirectSound interface to the TDirectSound object . The user can directly access the IDirectSound interface by using the ISound property.

See also

TDirectSoundBuffer

Hierarchy

TPersistent

TDirectX

TDirectSound properties

TDirectSound Lenqend

In TDirectSound

IDSound

ISound

Derived from TDirectX

DXResult

TDirectSound.IDSound

TDirectSound

```
property IDSound: IDirectSound;
```

Description

The property is used to access the IDirectSound interface directly . Even if a point different from the ISound property is nil,   the exception is not generated.

TDirectSound.ISound

TDirectSound

```
property ISound: IDirectSound;
```

Description

This property is used to access the IDirectSound interface directly.

TDirectSound methods

TDirectSound Lengend

In TDirectSound

Create

Destroy

Drivers

TDirectSound.Create

TDirectSound

```
constructor Create(GUID: PGUID);
```

Description

The TDirectSound object is made.

Argument	Explanation
GUID	The pointer to GUID of the DirectSound driver . When nil is specified, the driver of default is used.

See also

Drivers

TDirectSound.Destroy

TDirectSound

```
destructor Destroy;
```

Description

The TDirectSound object is abandoned.

TDirectSound.Drivers

TDirectSound

```
class function Drivers: TDirectXDrivers;
```

Description

The list of the DirectSound driver is returned.

TDirectSoundBuffer

Hierarchy Properties Methods

Unit

DXSounds

Description

The wrapping does the IDirectSoundBuffer interface to the TDirectSoundBuffer object . The user can directly access the IDirectSoundBuffer interface by using the IBuffer property .

To make the buffer, the SetSize method is called.

The Play method is called so that the reproduction of the sound buffer may begin . To stop reproducing, the Stop method is called.

Hierarchy

TPersistent

TDirectX

TDirectSoundBuffer properties

TDirectSoundBuffer Lengend

In TDirectSoundBuffer

BitCount

DSound

Frequency

IBuffer

IDSBuffer

Pan

Playing

Position

Status

Volume

Derived from TDirectX

DXResult

TDirectSoundBuffer.BitCount

TDirectSoundBuffer

```
property BitCount: Integer;
```

Description

It is a number of bits for each sample.

TDirectSoundBuffer.DSound

TDirectSoundBuffer

```
property DSound: TDirectSound;
```

Description

It is TDirectSound object.

TDirectSoundBuffer.Frequency

TDirectSoundBuffer

```
property Frequency: Integer;
```

Description

The reproduction frequency is specified within the range from 100 to 100000 . When 0 is specified, the value of default is set.

TDirectSoundBuffer.IBuffer

TDirectSoundBuffer

```
property IBuffer: Integer;
```

Description

This property is used to access the IDirectSoundBuffer interface directly.

TDirectSoundBuffer.IDSBuffer

TDirectSoundBuffer

```
property IDSBuffer: Integer;
```

Description

This property is used to access the IDirectSoundBuffer interface directly . Even if a point different from the IBuffer property is nil,   the exception is not generated.

TDirectSoundBuffer.Pan

TDirectSoundBuffer

```
property Pan: Integer;
```

Description

A relative volume of a right and left channel is specified within the range from -10000 to 10000.

A right channel is muted when -10000 is specified, and when 10000 is specified, a left channel is muted.

TDirectSoundBuffer.Playing

TDirectSoundBuffer

```
property Playing: Boolean;
```

Description

This property is returned whether reproducing now.

TDirectSoundBuffer.Position

TDirectSoundBuffer

```
property Position: Integer;
```

Description

The position which has reproduced now is specified by each byte.

TDirectSoundBuffer.Status

TDirectSoundBuffer

```
property Status: Integer;
```

Description

It is a state of the buffer.

Identifier	Meaning
DSBSTATUS_BUFFERLOST	The buffer is lost and must be restored before it can be played or locked.
DSBSTATUS_LOOPING	The buffer is being looped. If this value is not set, the buffer will stop when it reaches the end of the sound data. Note that if this value is set, the buffer must also be playing.
DSBSTATUS_PLAYING	The buffer is playing. If this value is not set, the buffer is stopped.

TDirectSoundBuffer.Volume

TDirectSoundBuffer

```
property Volume: Integer;
```

Description

The volume is specified within the range from -10000 to 0. It is muted that -10000 is specified,   and if 0 is specified,   becomes a usual volume.

TDirectSoundBuffer methods

TDirectSoundBuffer Lengend

In TDirectSoundBuffer

- Assign
- Create
- CreateBuffer
- Destroy
- GetFormat
- GetFormatAlloc
- LoadFromFile
- LoadFromMemory
- LoadFromStream
- LoadFromWave
- Lock
- Play
- Restore
- SetFormat
- SetSize
- Stop
- Unlock

TDirectSoundBuffer.Assign

TDirectSoundBuffer

```
procedure Assign(Source: TPersistent);
```

Description

The Assign method allocates the object compatible with TDirectSoundBuffer . The object at present compatible is TWave and TDirectSoundBuffer .

When the TDirectSoundBuffer object is allocated,   the Wave data is shared.

See also

TWave

TDirectSoundBuffer.Create

TDirectSoundBuffer

```
constructor Create(ADSound: TDirectSound);
```

Description

The TDirectSoundBuffer object is made.

Argument	Explanation
ADSound	The TDirectSound object . The value of the TDXSound.DSound property is specified.

TDirectSoundBuffer.CreateBuffer

TDirectSoundBuffer

```
function CreateBuffer(const ABufferDesc: DSBUFFERDESC): Boolean;
```

Description

The buffer is made . True is returned when succeeding.

Argument	Explanation
ABufferDesc	DSBUFFERDESC structural body which described information on made buffer

TDirectSoundBuffer.Destroy

TDirectSoundBuffer

```
destructor Destroy;
```

Description

The TDirectSoundBuffer object is abandoned.

TDirectSoundBuffer.GetFormat

TDirectSoundBuffer

```
function GetFormat(var Format: TWaveFormatEx; dwSizeAllocated: Longint; var
  dwSizeWritten: Longint): Boolean;
```

Description

The format of the buffer is acquired . True is returned when succeeding.

Argument	Explanation
Format	TWaveFormatEx structural body which receives format of buffer
dwSizeAllocated	Size of Format argument
dwSizeWriten	Size written in Format argument

TDirectSoundBuffer.GetFormatAlloc

TDirectSoundBuffer

```
function GetFormatAlloc(var Format: PWaveFormatEx; var Size: Longint):
  Boolean;
```

Description

The pointer to TWaveFormatEx structural body which described the format of the buffer is acquired . Please liberate the acquired pointer.

Argument	Explanation
Format	PWaveFormatEx variable by which pointer to TWaveFormatEx structural body which receives format of buffer is received
Size	Size for which acquired pointer is secured

TDirectSoundBuffer.LoadFromFile

TDirectSoundBuffer

```
procedure LoadFromFile(const FileName: string);
```

Description

The Wave file is read.

TDirectSoundBuffer.LoadFromMemory

TDirectSoundBuffer

```
procedure LoadFromMemory(const Format: TWaveFormatEx; Data: Pointer; Size:
  Integer);
```

Description

Wave is read.

Argument	Explanation
Format	TWaveFormatEx structural body which described format of read Wave
Data	Pointer to read Wave
Size	Size of read Wave

TDirectSoundBuffer.LoadFromStream

TDirectSoundBuffer

```
procedure LoadFromStream(Stream: TStream);
```

Description

Wave is read from the stream.

TDirectSoundBuffer.LoadFromWave

TDirectSoundBuffer

```
procedure LoadFromWave(Wave: TWave);
```

Description

The TWave object is read.

TDirectSoundBuffer.Lock

TDirectSoundBuffer

```
function Lock(dwWriteCursor, dwWriteBytes: Longint; var lpvAudioPtr1:
  Pointer; var dwAudioBytes1: Longint; var lpvAudioPtr2: Pointer; var
  dwAudioBytes2: Longint; dwFlags: Longint): Boolean;
```

Description

The buffer is locked . True is returned when succeeding.

Argument	Explanation
dwWriteCursor	Position of buffer where lock begins
dwWriteBytes	Locked size
lpvAudioPtr, dwAudioBytes	Pointer and locked size to memory of locked buffer
lpvAudioPtr2, dwAudioBytes2	Pointer and locked size 2 to memory 2 of locked buffer
dwFlags	The lock flag . The following one or the one to have harmonized in logic is specified. DSBLOCK_FROMWRITECURSOR Locks from the current write position, making a call to IDirectSoundBuffer::GetCurrentPosition unnecessary. If this flag is specified, the dwWriteCursor parameter is ignored.

See also

Unlock

TDirectSoundBuffer.Play

TDirectSoundBuffer

```
function Play(Flags: Longint): Boolean;
```

Description

The reproduction begins . True is returned when succeeding.

Argument	Explanation
Flags	The reproduction flag . The following one or the one to have harmonized in logic is specified. DSBPLAY_LOOPING Once the end of the audio buffer is reached, play restarts at the beginning of the buffer. Play continues until explicitly stopped. This flag must be set when playing primary sound buffers.

TDirectSoundBuffer.Restore

TDirectSoundBuffer

```
function Restore: Boolean;
```

Description

The buffer is restored . True is returned when succeeding.

TDirectSoundBuffer.SetFormat

TDirectSoundBuffer

```
function SetFormat(const Format: TWaveFormatEx): Boolean;
```

Description

The format of the buffer is set . True is returned when succeeding.

Argument	Explanation
Format	TWaveFormatEx structural body which described set format

TDirectSoundBuffer.SetSize

TDirectSoundBuffer

```
function SetSize(const Format: TWaveFormatEx; Size: Integer): Boolean;
```

Description

This method is made by the size specified the format by which the buffer is specified . True is returned when succeeding.

Argument	Explanation
Format	TWaveFormatEx structural body which described format of new buffer
Size	Size of new buffer

TDirectSoundBuffer.Stop

TDirectSoundBuffer

```
function Stop: Boolean;
```

Description

The reproduction is stopped . True is returned when succeeding.

TDirectSoundBuffer.Unlock

TDirectSoundBuffer

```
function Unlock(lpvAudioPtr1: Pointer; dwAudioBytes1: Longint; lpvAudioPtr2:
  Pointer; dwAudioBytes2: Longint): Boolean;
```

Description

The lock is released . True is returned when succeeding.

Argument	Explanation
lpvAudioPtr, dwAudioBytes	Pointer and locked size to memory of locked buffer
lpvAudioPtr2, dwAudioBytes2	Pointer and locked size 2 to memory 2 of locked buffer

See also

Lock

TSoundCaptureFormat

Hierarchy Properties

Unit

DXSounds

Description

The TSoundCaptureFormat object is a format of the sound which can be captured.

See also

TSoundCaptureFormats

Hierarchy

TCollectionItem

TSoundCaptureFormat properties

TSoundCaptureFormat Lengend

In TSoundCaptureFormat

- BitsPerSample
- Channels
- SamplesPerSec

TSoundCaptureFormat.BitsPerSample

TSoundCaptureFormat

```
property BitsPerSample: Integer;
```

Description

It is a number of bits for each sample.

TSoundCaptureFormat.Channels

TSoundCaptureFormat

```
property Channels: Integer;
```

Description

It is a number of channels . One is monaural,   and two is stereo.

TSoundCaptureFormat.SamplesPerSec

TSoundCaptureFormat

```
property SamplesPerSec: Integer;
```

Description

It is a frequency.

TSoundCaptureFormats

Hierarchy Properties Methods

Unit

DXSounds

Description

The TSoundCaptureFormats object is a list of the format of the sound which can be captured.

See also

TSoundCaptureStream

Hierarchy

TCollection

TSoundCaptureFormats properties

TSoundCaptureFormats   Lenqend

In TSoundCaptureFormats

Count

Items

TSoundCaptureFormats.Count

TSoundCaptureFormats

```
property Count: Integer;
```

Description

It is a number of formats being maintained now.

TSoundCaptureFormats.Items

TSoundCaptureFormats

```
property Items[Index: Integer]: TSoundCaptureFormat;
```

Description

It is a list of the format being maintained now.

TSoundCaptureFormats methods

TSoundCaptureFormats Lengend

In TSoundCaptureFormats

IndexOf

TSoundCaptureFormats.IndexOf

TSoundCaptureFormats

```
function IndexOf(ASamplesPerSec, ABitsPerSample, AChannels: Integer):
  Integer;
```

Description

The index of the specified format is returned . -1 is returned when not found.

TSoundCaptureStream

Hierarchy Properties Methods Events See also Example

Unit

DXSounds

Description

The TSoundCaptureStream object is Wave stream to which the sound is captured.

The TSoundCaptureStream object has achieved capturing for a long time by taking it out one by one as soon as seconds of the buffer how many are secured, and the Wave data which captures there collects. The BufferLength property specifies the length of the buffer.

The Wave format when capturing by the CaptureFormat property can be specified. The list of the Wave format which can be used can be acquired in the SupportedFormats property.

To begin capturing, the Start method is called. To stop, the Stop method is called.

Timing where the Wave data collects can know that the OnFilledBuffer event is used.

The method of reading the TStream object is used to read the capturing Wave data.

Hierarchy

TStream

TCustomWaveStream

TCustomWaveStream2

TSoundCaptureStream properties

TSoundCaptureStream Lengend

In TSoundCaptureStream

BufferLength

CaptureFormat

Capturing

SupportedFormats

Derived from TCustomWaveStream

FilledSize

Format

FormatSize

Size

TSoundCaptureStream.BufferLength

TSoundCaptureStream

```
property BufferLength: Integer;
```

Description

The size of the reception buffer is specified in each millisecond. 1000 is specified by default.

The TSoundCaptureStream object has achieved capturing for a long time by taking it out as soon as seconds of the buffer how many are secured, and the Wave data captured to it collects. This property specifies the length of the buffer.

Capturing becomes specification of a big number for this property and the road comes not to cut easily.

TSoundCaptureStream.CaptureFormat

TSoundCaptureStream

```
property CaptureFormat: Integer;
```

Description

The capturing format is specified . It is an index of the item of the SupportedFormats property.

TSoundCaptureStream.Capturing

TSoundCaptureStream

```
property Capturing: Boolean;
```

Description

It is returned whether to be capturing now.

TSoundCaptureStream.SupportedFormats

TSoundCaptureStream

```
property SupportedFormats: TSoundCaptureFormats;
```

Description

It is a list of the format which can be captured . The format of the high pitched sound quality is stored by going at the end of the list.

TSoundCaptureStream methods

TSoundCaptureStream Lengend

In TSoundCaptureStream

Create

Destroy

Drivers

Start

Stop

Derived from TCustomWaveStream

GetFilledSize

GetFormat

GetFormatSize

GetSize

ReadWave

SetFormatSize

SetPCMFormat

SetSize

WriteWave

TSoundCaptureStream.Create

TSoundCaptureStream

```
constructor Create(GUID: PGUID);
```

Description

The TSoundCaptureStream object is made.

Argument	Explanation
GUID	The pointer to GUID of the DirectSoundCapture driver . When nil is passed, the driver of default is used.

See also

Drivers

TSoundCaptureStream.Destroy

TSoundCaptureStream

```
destructor Destroy;
```

Description

The TSoundCaptureStream object is abandoned.

TSoundCaptureStream.Drivers

TSoundCaptureStream

```
class function Drivers: TDirectXDrivers;
```

Description

The list of the DirectSoundCapture driver is returned.

TSoundCaptureStream.Start

TSoundCaptureStream

```
procedure Start;
```

Description

Capturing is begun.

TSoundCaptureStream.Stop

TSoundCaptureStream

```
procedure Stop;
```

Description

Capturing is stopped.

TSoundCaptureStream events

TSoundCaptureStream Lenqend

In TSoundCaptureStream

OnFilledBuffer

TSoundCaptureStream.OnFilledBuffer

TSoundCaptureStream Example

```
property OnFilledBuffer: TNotifyEvent;
```

Description

It is an event generated when came to being able to read.

Example of TSoundCaptureStream.OnFilledBuffer

The Wave data that the OnFilledBuffer event's occurring captures to Dest is written.

```
var
  Capture: TSoundCaptureStream;
  Dest: TWaveFileStream;

procedure TForm1.CaptureStreamEvent(Sender: TObject);
var
  Capture: TSoundCaptureStream;
begin
  Capture := TSoundCaptureStream(Sender);
  Dest.CopyFrom(Capture, Capture.FilledSize);
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  Capture.OnFilledBuffer := CaptureStreamEvent;
  Capture.Start;
end;
```

Example of TSoundCaptureStream

The sound input from capturing is written in the file.

```
uses Wave, DXSounds;

const
  CaptureSec = 10; {   The sound is captured for ten seconds.   }

procedure TForm1.Button1Click(Sender: TObject);
var
  Stream: TSoundCaptureStream;
  WaveFileStream: TWaveFileStream;
begin
  Stream := TSoundCaptureStream.Create(nil);
  try
    {   The sound is captured by the highest tone quality.   }
    Stream.CaptureFormat := Stream.SupportedFormats.Count-1;

    {   Capturing of the sound is begun.   }
    Stream.Start;

    WaveFileStream := TWaveFileStream.Create('c:\test.wav', fmCreate);
    try
      WaveFileStream.FormatSize := Stream.FormatSize;
      Move(Stream.Format^, WaveFileStream.Format^,
WaveFileStream.FormatSize);

      WaveFileStream.Open(True);

      WaveFileStream.CopyFrom(Stream,

WaveFileStream.Format^.nSamplesPerSec*WaveFileStream.Format^.nBlockAlign*Capt
ureSec);
    finally
      WaveFileStream.Free;
    end;
  finally
    Stream.Free;
  end;
end;
```

TWaveCollection

Hierarchy Properties Methods Example

Unit

DXSounds

Description

It is a collection of TWave.

See also

TWaveCollectionItem

Hierarchy

TCollection

THashCollection

TWaveCollection properties

TWaveCollection Lengend

In TWaveCollection

Count

Items

TWaveCollection.Count

TWaveCollection

```
property Count: Integer;
```

Description

The number of Wave being maintained now is returned.

TWaveCollection.Items

TWaveCollection

```
property Items[Index: Integer]: TWaveCollectionItem; default;
```

Description

It is a list of Wave being maintained now.

TWaveCollection methods

TWaveCollection Lengend

In TWaveCollection

Clear

Create

Destroy

Find

IndexOf

LoadFromFile

LoadFromStream

Restore

SaveToFile

SaveToStream

TWaveCollection.Clear

TWaveCollection

```
procedure Clear;
```

Description

All Wave being maintained now is abandoned.

TWaveCollection.Create

TWaveCollection

```
constructor Create;
```

Description

The TWaveCollection object is made.

TWaveCollection.Destroy

TWaveCollection

```
destructor Destroy;
```

Description

The TWaveCollection object is abandoned.

TWaveCollection.Find

TWaveCollection

```
procedure Find(const Name: string): TWaveListItem;
```

Description

Wave of the specified name is returned . The hash is used and the operation is high-speed . When not found,   the exception is generated.

TWaveCollection.IndexOf

TWaveCollection

```
procedure IndexOf(const Name: string): Integer;
```

Description

The index of Wave of the specified name is returned . The hash is used and the operation is high-speed . When not found,   -1 is returned.

TWaveCollection.LoadFromFile

TWaveCollection

```
procedure LoadFromFile(const FileName: string);
```

Description

The Wave list is read from the file.

TWaveCollection.LoadFromStream

TWaveCollection

```
procedure SaveToStream(Stream: TStream);
```

Description

The Wave list is read from the stream.

TWaveCollection.Restore

TWaveCollection

```
procedure Restore;
```

Description

Wave is loaded into the buffer.

TWaveCollection.SaveToFile

TWaveCollection

```
procedure SaveToFile(const FileName: string);
```

Description

All Wave being maintained now is preserved in the file.

TWaveCollection.SaveToStream

TWaveCollection

```
procedure SaveToStream(Stream: TStream);
```

Description

All Wave being maintained now is preserved in the stream.

Example of TWaveCollection

New Wave is added to the collection.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  Item: TWaveCollectionItem;
begin
  Item := TWaveCollectionItem.Create(DXWaveList1.Items);
  Item.Name := 'New Wave';
  Item.Wave.LoadFromFile('c:\wavefile.wav');
end;
```

TWaveCollectionItem

Hierarchy Properties Methods

Unit

DXSounds

Description

The TWaveCollectionItem object is an item of the TWaveCollection object .

Wave can be specified with the Wave property .

To reproduce, the Play method is called. In that case, to loop, the Looped property is set in True.

Hierarchy

TCollectionItem

THashCollectionItem

TWaveCollectionItem properties

TWaveCollectionItem Lengend

In TWaveCollectionItem

Buffer

Frequency

Initialized

Looped

Name

Pan

Volume

Wave

TWaveCollectionItem.Buffer

TWaveCollectionItem

```
property Buffer: TDirectSoundBuffer;
```

Description

It is a buffer.

TWaveCollectionItem.Frequency

TWaveCollectionItem

```
property Frequency: Integer;
```

Description

The reproduction frequency is specified within the range from 100 to 100000 . When 0 is specified,　the value of default is set.

When the Looped property is True,　the change in this property is reflected in the output at once.

TWaveCollectionItem.Initialized

TWaveCollectionItem

```
property Initialized: Boolean;
```

Description

The buffer is being initialized now or this property is returned.

TWaveCollectionItem.Looped

TWaveCollectionItem

```
property Looped: Boolean;
```

Description

It is specified whether to do the loop reproduction.

TWaveCollectionItem.Name

TWaveCollectionItem

```
property Name: string;
```

Description

It is a name of this Wave.

TWaveCollectionItem.Pan

TWaveCollectionItem

```
property Pan: Integer;
```

Description

A relative volume of a right and left channel is specified within the range from -10000 to 10000.

A right channel is muted when -10000 is specified, and when 10000 is specified, a left channel is muted.

When the Looped property is True, the change in this property is reflected in the output at once.

TWaveCollectionItem.Volume

TWaveCollectionItem

```
property Volume: Integer;
```

Description

The volume is specified within the range from -10000 to 0. It is muted that -10000 is specified,　and if 0 is specified,　becomes a usual volume.

When the Looped property is True,　the change in this property is reflected in the output at once.

TWaveCollectionItem.Wave

TWaveCollectionItem

```
property Wave: TWave;
```

Description

It is TWave object. To reflect the content which Wave changed in the sound buffer,   the Restore method is called.

TWaveCollectionItem methods

TWaveCollectionItem Lengend

In TWaveCollectionItem

Play

Restore

Stop

TWaveCollectionItem.Play

TWaveCollectionItem

```
procedure Play(Wait: Boolean);
```

Description

The sound is reproduced . The sound is reproduced . The loop reproduction is done when the Looped property is True,　and the plural is reproduced simultaneously at False.

At this time,　the value of the Frequency, Pan and Volume property is used.

Argument	Explanation
Wave	This argument specifies whether to wait until the reproduction ends.

TWaveCollectionItem.Restore

TWaveCollectionItem

```
procedure Restore;
```

Description

Wave is loaded into the buffer.

TWaveCollectionItem.Stop

TWaveCollectionItem

```
procedure Stop;
```

Description

The loop reproduction is stopped.

DXSprite unit

The class concerning Sprite is declared to the DXSprite unit.
The following item is declared to the DXSprite unit.

Components

TDXSpriteEngine

Objects

TBackgroundSprite

TImageSprite

TSprite

TSpriteEngine

TDXSpriteEngine

Hierarchy Properties Methods

Unit

DXSprite

Description

The TDXSpriteEngine component controls sprite .

To do rendering, the Render method is called . The DXDraw property is used to set rendering ahead.

Hierarchy

TComponent

TCustomDXSpriteEngine

TDXSpriteEngine properties

TDXSpriteEngine Lengend

In TDXSpriteEngine

DXDraw

Engine

TDXSpriteEngine.DXDraw

TDXSpriteEngine

```
property DXDraw: TDXDraw;
```

Description

The TDXDraw component the description ahead is specified.

TDXSpriteEngine.Engine

TDXSpriteEngine

```
property Engine: TSpriteEngine;
```

Description

It is TSpriteEngine object.

TDXSpriteEngine methods

TDXSpriteEngine Lengend

In TDXSpriteEngine

Dead

Move

Render

TDXSpriteEngine.Dead

TDXSpriteEngine

```
procedure Dead;
```

Description

Sprite with the deletion demand is deleted.

TDXSpriteEngine.Move

TDXSpriteEngine

```
procedure Move(MoveCount: Integer);
```

Description

Sprite is moved.

Argument	Explanation
MoveCount	The moving time is specified in each millisecond.

TDXSpriteEngine.Render

TDXSpriteEngine

```
procedure Render;
```

Description

Sprite is described.

TBackgroundSprite

Hierarchy Properties Methods

Unit

DXSprite

Description

The TBackground object is sprite chiefly used for the background.

Which chip of each part is described with the Chips property can be specified.

Effective and the invalidity of each part of the collision judgment can be set with the CollisionMap property.

Hierarchy

TSprite

TBackgroundSprite properties

TBackgroundSprite Lengend

In TBackgroundSprite

Chips

CollisionMap

Image

MapHeight

MapWidth

Tile

Derived from TSprite

BoundsRect

ClientRect

Collisioned

Count

Engine

Height

Items

Moved

Parent

Visible

Width

WorldX

WorldY

X

Y

Z

TBackgroundSprite.Chips

TBackgroundSprite

```
property Chips[X, Y: Integer]: Integer;
```

Description

It is a pattern number of the image of the chip . The chip is not described to specify a negative number.

TBackgroundSprite.CollisionMap

TBackgroundSprite

```
property CollisionMap[X, Y: Integer]: Boolean;
```

Description

It can be set whether to do the collision judgment in each chip when the CollisionMap property is used.

TBackgroundSprite.Image

TBackgroundSprite

```
property Image: TPictureCollectionItem;
```

Description

It is an image of the chip.

TBackgroundSprite.MapHeight

TBackgroundSprite

```
property MapHeight: Integer;
```

Description

It is height of the map.

TBackgroundSprite.MapWidth

TBackgroundSprite

```
property MapWidth: Integer;
```

Description

It is width of the map.

TBackgroundSprite.Tile

TBackgroundSprite

```
property Tile: Boolean;
```

Description

It is specified whether to always cover the screen . It is always described that True is set on the screen.

TBackgroundSprite methods

TBackgroundSprite Lenqend

In TBackgroundSprite

SetMapSize

Derived from TSprite

Clear

Collision

Create

Dead

Destroy

DoCollision

DoDraw

DoMove

GetBoundsRect

TestCollision

TBackgroundSprite.SetMapSize

TBackgroundSprite

```
property SetMapSize(AMapWidth, AMapHeight: Integer): Integer;
```

Description

The size of the map is set.

TImageSprite

Hierarchy Properties Methods

Unit

DXSprite

Description

The TImageSprite object is sprite by which the image is displayed.

The described image is specified with the Image property . The described surface is Image.PatternSurfaces[AnimStart+AnimPos].

Hierarchy

TSprite

TImageSprite properties

TImageSprite Lenqend

In TImageSprite

AnimCount

AnimLooped

AnimPos

AnimSpeed

AnimStart

Image

PixelCheck

Tile

Derived from TSprite

BoundsRect

ClientRect

Collisioned

Count

Engine

Height

Items

Moved

Parent

Visible

Width

WorldX

WorldY

X

Y

Z

TImageSprite.AnimCount

TImageSprite Example

```
property AnimCount: Integer;
```

Description

It is a number of patterns of animations.

Example of TImageSprite.AnimXXX

```
begin
  ImageSprite.AnimStart := 10;      {    Animation is reproduced from the pattern number
10.   }
  ImageSprite.AnimLooped := True;   {    Loop    }
  ImageSprite.AnimSpeed := 10/1000; {    They are ten frames a second.     }
  ImageSprite.AnimCount := 3;       {    The number of patterns is three.     }
end;
```

TImageSprite.AnimLooped

TImageSprite Example

```
property AnimLooped: Boolean;
```

Description

It is specified whether to loop and to do animation. When animation is ended if False is set, 0 is substituted for the AnimSpeed property.

TImageSprite.AnimPos

TImageSprite

```
property AnimPos: Double;
```

Description

It is a relative position of the AnimStart property of the pattern number of animation.

TImageSprite.AnimSpeed

TImageSprite Example

```
property AnimSpeed: Double;
```

Description

It is a speed of animation . Whether animation is done is specified of what frame millisecond.

TImageSprite.AnimStart

TImageSprite Example

```
property AnimStart: Integer;
```

Description

It is a pattern number by which animation is started.

TImageSprite.Image

TImageSprite

```
property Image: TPictureCollectionItem;
```

Description

It is a described image.

TImageSprite.PixelCheck

TImageSprite

```
property PixelCheck: Boolean;
```

Description

It is specified whether to judge each pixel when other TImageSprite objects and the collision judgments are done . When False is specified,   the collision judgment is judged in the rectangle.

TImageSprite.Tile

TImageSprite

```
property Tile: Boolean;
```

Description

It is specified whether to display sprite without fail regardless of the position.

TImageSprite methods

TImageSprite Lengend

Derived from TSprite

Clear

Collision

Create

Dead

Destroy

DoCollision

DoDraw

DoMove

GetBoundsRect

TestCollision

TSprite

Hierarchy Properties Methods

Unit

DXSprite

Description

The TSprite object is an abstraction class of sprite . Sprite can be constructed in the tree, and some properties are influenced by the parent.

When the description processing is done, the DoDraw method is overrided.
When the movement processing is done, the DoMove method is overrided.
To do the collision judgment, the Collision method is called . When it is processed to collide, the DoCollision method is overrided.

Hierarchy

TObject

TSprite properties

TSprite Lengend

In TSprite

- BoundsRect
- ClientRect
- Collisioned
- Count
- Engine
- Height
- Items
- Moved
- Parent
- Visible
- Width
- WorldX
- WorldY
- X
- Y
- Z

TSprite.BoundsRect

TSprite

```
property BoundsRect: TRect;
```

Description

The rectangle in the world coordinates is returned . This property is used to describe and to judge the collision.

TSprite.ClientRect

TSprite

```
property ClientRect: TRect;
```

Description

The rectangle of sprite is returned . It is the same as Rect(0, 0, Width, Height).

TSprite.Collisioned

TSprite

```
property Collisioned: Boolean;
```

Description

It is specified whether to do the collision judgment.

TSprite.Count

TSprite

```
property Count: Integer;
```

Description

The number of sprite being maintained now is returned.

TSprite.Engine

TSprite

```
property Engine: TSpriteEngine;
```

Description

It is TSpriteEngine object.

TSprite.Height

TSprite

```
property Height: Integer;
```

Description

It is height of sprite.

TSprite.Items

TSprite

```
property Items[Index: Integer]: TSprite; default;
```

Description

It is a list of the sprite being maintained now.

TSprite.Moved

TSprite

```
property Moved: Boolean;
```

Description

It is specified whether to move sprite.

TSprite.Parent

TSprite

```
property Parent: TSprite;
```

Description

It is parent's sprite.

TSprite.Visible

TSprite

```
property Visible: Boolean;
```

Description

It is specified whether to display sprite.

TSprite.Width

TSprite

```
property Width: Integer;
```

Description

It is width of sprite.

TSprite.WorldX

TSprite

```
property WorldX: Double;
```

Description

It is X coordinates in the world coordinates of sprite.

TSprite.WorldY

TSprite

```
property WorldY: Double;
```

Description

It is Y coordinates in the world coordinates of sprite.

TSprite.X

TSprite

```
property X: Double;
```

Description

It is X coordinates of sprite.

TSprite.Y

TSprite

```
property Y: Double;
```

Description

It is Y coordinates of sprite.

TSprite.Z

TSprite

```
property Z: Integer;
```

Description

It is Z value of sprite . Sprite is displayed forward by the value large.

TSprite methods

TSprite Lengend

In TSprite

- Clear
- Collision
- Create
- Dead
- Destroy
- DoCollision
- DoDraw
- DoMove
- GetBoundsRect
- TestCollision

TSprite.Clear

TSprite

```
procedure Clear;
```

Description

All the sprite being maintained now is abandoned.

TSprite.Collision

TSprite

```
procedure Collision;
```

Description

The collision judgment is done . When the sprite which has collided is detected,   the DoCollision method is called one by one.

TSprite.Create

TSprite

```
constructor Create(AParent: TSprite);
```

Description

The TSprite object is made.

Argument	Explanation
AParent	Parent's sprite

TSprite.Dead

TSprite

```
procedure Dead;
```

Description

The deletion demand is lodged . Sprite is completely deleted by the TSpriteEngine.Dead method.

TSprite.Destroy

TSprite

```
destructor Destroy;
```

Description

The TSprite object is abandoned.

TSprite.DoCollision

TSprite

```
procedure DoCollision(Sprite: TSprite; var Done: Boolean); virtual;
```

Description

It is processed to collide with other sprite.

Argument	Explanation
Sprite	Colliding sprite
Done	To interrupt the detection of the colliding sprite, True is substituted for the Done argument.

TSprite.DoDraw

TSprite

```
procedure DoDraw; virtual;
```

Description

It is processed to describe sprite.

TSprite.DoMove

TSprite

```
procedure DoMove(MoveCount: Integer); virtual;
```

Description

It is processed to move sprite.

Argument	Explanation
MoveCount	The moving time is specified in each millisecond.

TSprite.GetBoundsRect

TSprite

```
function GetBoundsRect: TRect; virtual;
```

Description

The value of the BoundsRect property is defined.

TSprite.TestCollision

TSprite

```
function TestCollision(Sprite: TSprite): Boolean; virtual;
```

Description

This method returns whether to have collided with the sprite given by the Sprite argument.

TSpriteEngine

Hierarchy Properties Methods

Unit

DXSprite

Description

The TSpriteEngine object controls sprite.

Hierarchy

TObject

TSprite

TSpriteEngine properties

TSpriteEngine   Lengend

In TSpriteEngine

AllCount

DrawCount

Surface

SurfaceRect

Derived from TSprite

BoundsRect

ClientRect

Collisioned

Count

Engine

Height

Items

Moved

Parent

Visible

Width

WorldX

WorldY

X

Y

Z

TSpriteEngine.AllCount

TSpriteEngine

```
property AllCount: Integer;
```

Description

It is a number of sprite.

TSpriteEngine.DrawCount

TSpriteEngine

```
property DrawCount: Integer;
```

Description

It is a number of described sprite.

TSpriteEngine.Surface

TSpriteEngine

```
property Surface: TDirectDrawSurface;
```

Description

It is a surface the description ahead.

TSpriteEngine.SurfaceRect

TSpriteEngine

```
property SurfaceRect: TRect;
```

Description

It is a rectangle of the surface.

TSpriteEngine methods

TSpriteEngine   Lenqend

In TSpriteEngine

Create

Dead

Destroy

Draw

Move

Derived from TSprite

Clear

Collision

Create

Dead

Destroy

DoCollision

DoDraw

DoMove

GetBoundsRect

TestCollision

TSpriteEngine.Create

TSpriteEngine

```
constructor Create;
```

Description

The TSpriteEngine object is made.

TSpriteEngine.Dead

TSpriteEngine

```
procedure Dead;
```

Description

Sprite with the deletion demand is deleted.

TSpriteEngine.Destroy

TSpriteEngine

```
destructor Destroy;
```

Description

The TSpriteEngine object is abandoned.

TSpriteEngine.Draw

TSpriteEngine

```
procedure Draw(Surface: TDirectDrawSurface);
```

Description

Sprite is described.

Argument	Explanation
Surface	Surface description ahead

TSpriteEngine.Move

TSpriteEngine

```
procedure Move(MoveCount: Integer);
```

Description

Sprite is moved.

Argument	Explanation
MoveCount	The moving time is specified in each millisecond.

Wave unit

The class concerning the Wave file is declared to the Wave unit.
The following item is declared to the Wave unit.

Components

TDXWave

Objects

TCustomWaveStream

TCustomWaveStream2

TWave

TWaveFileStream

TWaveObjectStream

TWaveStream

Routines

MakePCMWaveFormatEx

TDXWave

Hierarchy Properties

Unit

Wave

Description

The TDXWave component maintains Wave . The Wave property is used to access Wave.

Hierarchy

TComponent

TCustomDXWave

TDXWave properties

TDXWave Lenqend

In TDXWave

Wave

TDXWave.Wave

TDXWave

```
property Wave: TWave;
```

Description

It is TWave object.

TCustomWaveStream

Hierarchy Properties Methods

Unit

Wave

Description

The TCustomWaveStream object is an abstraction class of the Wave stream.

Hierarchy

TStream

TCustomWaveStream properties

TCustomWaveStream Lenqend

In TCustomWaveStream

- FilledSize
- Format
- FormatSize
- Size

TCustomWaveStream.FilledSize

TCustomWaveStream

```
property FilledSize: Integer read GetFilledSize;
```

Description

The size which can be read now is returned . When the size which can be read is fixed,   -1 is returned.

TCustomWaveStream.Format

TCustomWaveStream

```
property Format: PWaveFormatEx read GetFormat;
```

Description

When the Format property is used, information on Wave is obtained . The Format property is secured by the size returned by the FormatSize property.

TCustomWaveStream.FormatSize

TCustomWaveStream

```
property FormatSize: Integer read GetFormatSize write SetFormatSize;
```

Description

It is a size for which the Format property is secured.

TCustomWaveStream.Size

TCustomWaveStream

```
property Size: Integer read GetSize write SetSize;
```

Description

It is a size of Wave . When -1 is specified,   the size of the stream is regarded as infinity.

TCustomWaveStream methods

TCustomWaveStream Lengend

In TCustomWaveStream

- GetFilledSize
- GetFormat
- GetFormatSize
- GetSize
- ReadWave
- SetFormatSize
- SetPCMFormat
- SetSize
- WriteWave

TCustomWaveStream.GetFilledSize

TCustomWaveStream

```
function GetFilledSize: Integer; virtual;
```

Description

The value of the FilledSize property is defined.

TCustomWaveStream.GetFormat

TCustomWaveStream

```
function GetFormat: PWaveFormatEx; virtual; abstract;
```

Description

The value of the Format property is defined.

TCustomWaveStream.GetFormatSize

TCustomWaveStream

```
function GetFormatSize: Integer; virtual;
```

Description

The value of the FormatSize property is defined.

TCustomWaveStream.GetSize

TCustomWaveStream

```
function GetSize: Integer; virtual;
```

Description

The value of the Size property is defined.

TCustomWaveStream.ReadWave

TCustomWaveStream

```
function ReadWave(const Buffer; Count: Integer): Integer; virtual;
```

Description

Wave is read from the position of the Position property to Buffer . And,   the read size is returned.

TCustomWaveStream.SetFormatSize

TCustomWaveStream

```
procedure SetFormatSize(Value: Integer); virtual; abstract;
```

Description

When the value is set in the Format property,   this method is called.

TCustomWaveStream.SetPCMFormat

TCustomWaveStream

```
procedure SetPCMFormat(SamplesPerSec, BitsPerSample, Channels: Integer);
```

Description

The format of Wave is set in PCM .

For instance,　to make the format of 44100Hz 16bit stereo,　it is assumed SetPCMFormat(44100, 16, 2).

Argument	Explanation
SamplesPerSec	Frequency
BitsPerSample	The number of bits used for about one sample . 8 and 16 are specified.
Channels	The number of channels . One becomes in monaural and two becomes stereos.

TCustomWaveStream.SetSize

TCustomWaveStream

```
procedure SetSize(Value: Integer); virtual;
```

Description

The size of the data of Wave is set.

TCustomWaveStream.WriteWave

TCustomWaveStream

```
function WriteWave(var Buffer; Count: Integer): Integer; virtual;
```

Description

Buffer is written from the position of the Position property in Wave . And,   the written size is returned.

TCustomWaveStream2

Hierarchy Properties Methods

Unit

Wave

Description

It is the one that the GetFormat method of the TCustomWaveStream object, the GetFormatSize method, and the SetFormatSize method were mounted.

Hierarchy

TStream

TCustomWaveStream

TCustomWaveStream2 properties

TCustomWaveStream2 Lenqend

Derived from TCustomWaveStream

FilledSize

Format

FormatSize

Size

TCustomWaveStream2 methods

TCustomWaveStream2 Lengend

Derived from TCustomWaveStream

GetFilledSize

GetFormat

GetFormatSize

GetSize

ReadWave

SetFormatSize

SetPCMFormat

SetSize

WriteWave

TWave

Hierarchy Properties Methods

Unit

Wave

Description

The TWave object maintains Wave .

Information on Wave is obtained by the Format property . The Data property is used to access the Wave.

Hierarchy

TPersistent

TWave properties

TWave Lenqend

In TWave

Data

Format

FormatSize

Size

TWave.Data

TWave

```
property Data: Pointer;
```

Description

The user can directly access Wave by using the Data property . The Data property is secured by the value of the Size property.

TWave.Format

TWave

```
property Format: PWaveFormatEx;
```

Description

It is a format of Wave . The Format property is secured by the value of the FormatSize property.

TWave.FormatSize

TWave

```
property FormatSize: Integer;
```

Description

It is a size for which the Format property is secured.

TWave.Size

TWave

```
property Size: Integer;
```

Description

It is a size of Wave . There need to be the multiple of Format.nBlockAlign set value.

TWave methods

TWave   Lengend

In TWave

- Assign
- Clear
- Create
- Destroy
- LoadFromFile
- LoadFromStream
- SaveToFile
- SaveToStream
- SetPCMFormat

TWave.Assign

TWave

```
procedure Assign(Source: TPersistent);
```

Description

The object compatible with TWave is allocated . The object at present compatible is only TWave object.

TWave.Clear

TWave

```
procedure Clear;
```

Description

Wave being maintained now is abandoned.

TWave.Create

TWave

```
constructor Create;
```

Description

The TWave object is made.

TWave.Destroy

TWave

```
destructor Destroy;
```

Description

The TWave object is abandoned.

TWave.LoadFromFile

TWave

```
procedure LoadFromFile(const FileName: string);
```

Description

Wave is read from the file.

TWave.LoadFromStream

TWave

```
procedure LoadFromStream(Stream: TStream);
```

Description

Wave is read from the stream.

TWave.SaveToFile

TWave

```
procedure SaveToFile(const FileName: string);
```

Description

Wave is written in the file.

TWave.SaveToStream

TWave

```
procedure SaveToStream(Stream: TStream);
```

Description

Wave is written in the stream.

TWave.SetPCMFormat

TWave

```
procedure SetPCMFormat(SamplesPerSec, BitsPerSample, Channels: Integer);
```

Description

The format of Wave is set in PCM.

For instance, to make the format of 44100Hz 16bit stereo, it is assumed SetPCMFormat(44100, 16, 2).

Argument	Explanation
SamplesPerSec	Frequency
BitsPerSample	The number of bits used for about one sample . 8 and 16 are specified.
Channels	The number of channels . One becomes in monaural and two becomes stereos.

TWaveFileStream

Hierarchy Properties Methods

Unit

Wave

Description

The TWaveFileStream object is Wave stream by which the Wave file is treated.
Some properties and the methods cannot be used until the Open method is called.

Hierarchy

TStream

TCustomWaveStream

TCustomWaveStream2

TWaveStream

TWaveFileStream properties

TWaveFileStream Lengend

Derived from TCustomWaveStream

FilledSize

Format

FormatSize

Size

TWaveFileStream methods

TWaveFileStream Lengend

In TWaveFileStream

Create

Derived from TWaveStream

Create

Open

Derived from TCustomWaveStream

GetFilledSize

GetFormat

GetFormatSize

GetSize

ReadWave

SetFormatSize

SetPCMFormat

SetSize

WriteWave

TWaveFileStream.Create

TWaveFileStream Example

```
constructor Create(const AFileName: string; Mode: Word);
```

Description

The TWaveObjectStream object can treat the TWave object as Wave stream.

Argument	Explanation
AFileName	Wave file name
Mode	The mode when the file is opened . Please refer to the explanation of the TFileStream.Create method.

TWaveObjectStream

Hierarchy Properties Methods

Unit

Wave

Description

The TWaveObjectStream object can treat the TWave object as Wave stream.

Hierarchy

TStream

TCustomWaveStream

TWaveObjectStream properties

TWaveObjectStream Lengend

Derived from TCustomWaveStream

FilledSize

Format

FormatSize

Size

TWaveObjectStream methods

TWaveObjectStream Lengend

In TWaveObjectStream

Create

Derived from TCustomWaveStream

GetFilledSize

GetFormat

GetFormatSize

GetSize

ReadWave

SetFormatSize

SetPCMFormat

SetSize

WriteWave

TWaveObjectStream.Create

TWaveObjectStream

```
constructor Create(AWave: TWave);
```

Description

The TWaveObjectStream object is made.

Argument	Explanation
AWave	TWave object which wants to treat as Wave stream

TWaveStream

Hierarchy Properties Methods Example

Unit

Wave

Description

The TWaveStream object can treat the stream as Wave stream .
Some properties and the methods cannot be used until the Open method is called.

See also

TWaveFileStream

Hierarchy

TStream

TCustomWaveStream

TCustomWaveStream2

TWaveStream properties

TWaveStream Legend

Derived from TCustomWaveStream

FilledSize

Format

FormatSize

Size

TWaveStream methods

TWaveStream   Lengend

In TWaveStream

- Create
- Open

Derived from TCustomWaveStream

- GetFilledSize
- GetFormat
- GetFormatSize
- GetSize
- ReadWave
- SetFormatSize
- SetPCMFormat
- SetSize
- WriteWave

TWaveStream.Create

TWaveStream Example

```
constructor Create(AStream: TStream);
```

Description

The TWaveStream object is made . Please call the Open method after this.

Argument	Explanation
AStream	Stream which wants to treat as Wave stream

TWaveStream.Open

TWaveStream Example

```
procedure Open(OpenMode: Boolean);
```

Description

The Wave stream is opened.

Information can be read from the stream when opening for reading, and Wave be read.

Information can be written in the stream when the Open method is called open for writing, and Wave be written . At this time, information before is lost.

Argument	Explanation
OpenMode	The opening mode . When True is specified, the stream is opened for writing . When False is specified, the stream is opened for reading.

Example of TWaveStream

The Wave file is opened.

```
uses Wave;

procedure TForm1.Button1Click(Sender: TObject);
var
  Stream: TWaveFileStream;
begin
  Stream := TWaveFileStream.Create('c:\windows\media\The Microsoft
Sound.wav', fmOpenRead);
  try
    {   The stream is opened for reading.   }
    Stream.Open(False);
    {   Some processing   }
  finally
    Stream.Free;
  end;
end;
```

The noise of one second is made.

```
uses Wave, MMSystem;

procedure TForm1.Button1Click(Sender: TObject);
var
  Stream: TWaveFileStream;
  i: Integer;
  d: Byte;
begin
  Stream := TWaveFileStream.Create('c:\test.wav', fmCreate);
  try
    {   Format is set without fail before the Open method is called.   }
    {  22050Hz, 8bit, Mono  }
    Stream.SetPCMFormat(22050, 8, 1);

    {   The stream is opened for writing.   }
    Stream.Open(True);

    {   The noise is written for one second.   }
    for i:=0 to 22050-1 do
    begin
      d := Random(256);
      Stream.WriteBuffer(d, 1);
    end;
  finally
    Stream.Free;
  end;
end;
```

MakePCMWaveFormatEx function

Unit

Wave

Declaration

```
function MakePCMWaveFormatEx(var Format: TWaveFormatEx; SamplesPerSec,
  BitsPerSample, Channels: Integer);
```

Description

TWaveFormatEx structural body of PCM is made by given information.

For instance,　to make the format of 44100Hz 16bit stereo,　it is assumed MakePCMWaveFormatEx(Format, 44100, 16, 2).

Argument	Explanation
Format	The storage of the made format ahead
SamplesPerSec	Frequency
BitsPerSample	The number of bits used for about one sample . 8 and 16 are specified.
Channels	The number of channels . One becomes in monaural and two becomes stereos.

Programming Tips & FAQ

The technique when DelphiX is used is introduced here.

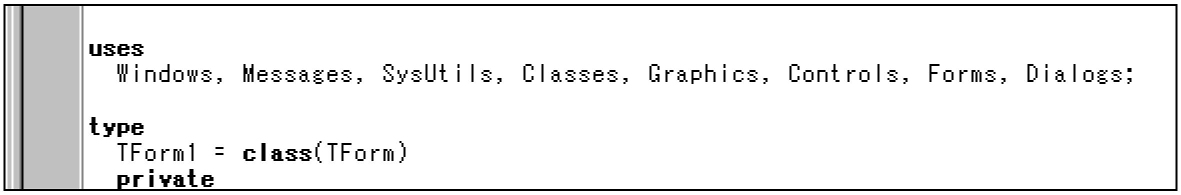
- Form is transformed to TDXForm.
- Screen mode of TDXDraw is switched.
- Palette animation is done with TDXDraw.
- Image is displayed with TDXImageList.
- OffScreenSurface is controlled for myself.
- The reproduction quality of TDXSound is set.

Form is transformed to TDXForm.

Programming Tips

Something wrong might occur when the TDXDraw component is used in TForm of the Delphi standard .
TDXForm to make something wrong canceled in DelphiX is prepared.

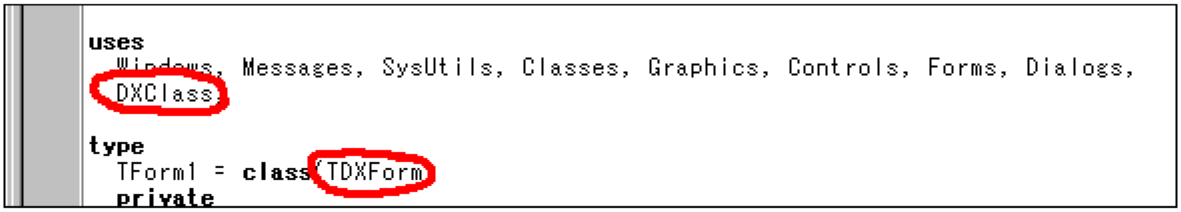
TForm is done and to be transformed of TDXForm,　as follows is done.



```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs;

type
  TForm1 = class(TForm)
  private
```

The DXClass unit is added to the Uses section,　and it is mended that it is TForm with TDXForm.



```
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  DXClass;

type
  TForm1 = class(TDXForm)
  private
```

Screen mode of TDXDraw is switched.

Programming Tips

If the button is pushed with TDXDraw,　the screen mode is switched.

Please refer to Form is transformed to TDXForm. declared in TDXForm for the RestoreWindow method and the StoreWindow method.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  DXDraw1.Finalize;

  if doFullScreen in DXDraw1.Options then
  begin
    {   From the full screen mode to the window mode    }
    RestoreWindow; {  The state of the window is restored.  }

    BorderStyle := bsSingle;
    DXDraw1.Cursor := crDefault;
    DXDraw1.Options := DXDraw1.Options - [doFullScreen];
  end else
  begin
    {   From the window mode to the full screen mode    }
    StoreWindow; {  The state of the window is preserved.  }

    BorderStyle := bsNone;
    DXDraw1.Cursor := crNone;
    DXDraw1.Options := DXDraw1.Options + [doFullScreen];

    {   There is a possibility that the content of the Display property has changed.    }
    with DXDraw1.Display do
    begin
      Width := 640;
      Height := 480;
      BitCount := 8;
    end;
  end;

  DXDraw1.Initialize;
end;
```

Palette animation is done with TDXDraw.

Programming Tips

The flash of the game and Fade in and Fade out, etc. are achieved by using the palette animation . However, the display mode can be used only at 256 color time because this technique uses the palette.

The CanPaletteAnimation property is checked without fail before the palette animation is done.

1. Fade out

The Fade out processing can lighten gradually, darken, and this be achieved by gradually bringing the palette close from a usual palette to the black and white.

Fade out is done to the color specified by the Col argument in the time specified by the Time argument.

```
function ComposeColor(Dest, Src: TRGBQuad; Percent: Integer): TRGBQuad;
begin
  with Result do
  begin
    rgbRed := Src.rgbRed+((Dest.rgbRed-Src.rgbRed)*Percent div 256);
    rgbGreen := Src.rgbGreen+((Dest.rgbGreen-Src.rgbGreen)*Percent div 256);
    rgbBlue := Src.rgbBlue+((Dest.rgbBlue-Src.rgbBlue)*Percent div 256);
    rgbReserved := 0;
  end;
end;

procedure TForm1.FadeOut(Time: Integer; Col: Integer);
begin
  {  Whether it is allowed to describe with TDXDraw is checked.  }
  if not DXDraw1.CanDraw then Exit;

  {  It is checked whether can do the palette animation with TDXDraw.   }
  if not DXDraw1.CanPaletteAnimation then Exit;

  {  Fade out  }
  t := GetTickCount;
  o := 0;
  while Abs(GetTickCount-t)<Time do
  begin
    p := Min(Max(Abs(GetTickCount-t)*255 div Time, 0), 255);

    if p<>o then
    begin
```

```
      o := p;

      for i:=0 to 255 do
        DXDraw1.ColorTable[i] := ComposeColor(RGBQuad(GetRValue(Col),
GetGValue(Col), GetBValue(Col)),
          DXDraw1.DefColorTable[i], p);

      DXDraw1.UpdatePalette;
    end;
  end;
end;
```

2. Fade in

The Fade in processing gradually returns to a usual palette . This can be achieved by gradually returning the palette from the black and the color to a usual palette.

Fade in is done from the color specified by the Col argument in the time specified by the Time argument.

```
procedure TForm1.FadeOut(Time: Integer; Col: Integer);
begin
  {  Whether it is allowed to describe with TDXDraw is checked.  }
  if not DXDraw1.CanDraw then Exit;

  {  It is checked whether can do the palette animation with TDXDraw.  }
  if not DXDraw1.CanPaletteAnimation then Exit;

  {  Fade in  }
  t := GetTickCount;
  o := 0;
  while Abs(GetTickCount-t)<Time do
  begin
    p := 255-Min(Max(Abs(GetTickCount-t)*255 div Time, 0), 255);

    if p<>o then
    begin
      o := p;

      for i:=0 to 255 do
        DXDraw1.ColorTable[i] := ComposeColor(RGBQuad(GetRValue(Col),
GetGValue(Col), GetBValue(Col)),
          DXDraw1.DefColorTable[i], p);

      DXDraw1.UpdatePalette;
    end;
  end;
end;
```

3. Flash

The flash processing can be achieved by first doing Fade out and doing Fade in next.

```
procedure TForm1.Flash(Time: Integer; Col: Integer);
begin
  FadeOut(Time div 2, Col);
  FadeIn(Time div 2, Col);
end;
```

Image is displayed with TDXImageList.

Programming Tips

The surface can be easily used by using the TDXImageList component.
It is necessary to set the DXDraw property and the Items property to use the TDXImageList component.

1. The palette is set.

It is used because there is a method of making a common palette in the TDXImageList component.
The palette is set here at a form OnCreate event.

```
procedure TForm1.Form1Create(Sender: TObject);
begin
  DXImageList1.Items.MakeColorTable;
  DXDraw1.ColorTable := DXImageList1.Items.ColorTable;
  DXDraw1.DefColorTable := DXImageList1.Items.ColorTable;
end;
```

2. Description which uses TDXImageList

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  {   Whether it is allowed to describe with TDXDraw is checked.   }
  if not DXDraw1.CanDraw then Exit;

  {   Pattern 0 of image 'Image' is described at a random position.   }
  DXImageList1.Items.Find('image').Draw(DXDraw1.Surface,
Random(DXDraw1.Surface.Width), Random(DXDraw1.Surface.Height), 0);

  {   The changed content is reflected in the screen.   }
  DXDraw1.Flip;
end;
```

OffScreenSurface is controlled for myself.

Programming Tips

The TDirectDrawSurface object is controlled for myself without using the TDXImageList component.

The TDirectDrawSurface object should do making,　abandonment,　and loading for myself unlike the TDXImageList component.

1. The surface is generated and abandoned.

Please add FSurface to form Private.

```
type
  TForm1 = class(TDXForm)
  private
    FSurface: TDirectDrawSurface;
  end;
```

The surface is generated because of the OnInitialize event.

```
procedure TForm1.DXDraw1Initialize(Sender: TObject);
begin
  FSurface := TDirectDrawSurface.Create(DXDraw1.DDraw);
end;
```

Moreover,　the surface generated because of the OnFinalize event is abandoned.

```
procedure TForm1.DXDraw1Finalize(Sender: TObject);
begin
  FSurface.Free;
  FSurface := nil;
end;
```

2. Contents of the surface are set.

Contents of the surface are set because of the OnRestoreSurface event . Here,　the noise is set in contents of the surface.

```
procedure TForm1.DXDraw1RestoreSurface(Sender: TObject);
var
  x, y: Integer;
begin
```

```
  {   The size of the surface is set in 64x64 pixel.   }
  FSurface.SetSize(64, 64);

  {   The noise is drawn.   }
  for y:=0 to FSurface.Height-1 do
    for x:=0 to FSurface.Width-1 do
      FSurface.Pixels[x, y] := Random(MaxInt);
end;
```

3. The surface is described.

```
procedure TForm1.Timer1Timer(Sender: TObject);
begin
  {   Whether it is allowed to describe with TDXDraw is checked.   }
  if not DXDraw1.CanDraw then Exit;

  DXDraw1.Surface.Fill(0);

  {   The surface is described at a random position.   }
  DXDraw1.Surface.Draw(Random(DXDraw1.Surface.Width),
Random(DXDraw1.Surface.Height), FSurface.ClientRect, FSurface, False);

  {   The changed content is reflected in the screen.   }
  DXDraw1.Flip;
end;
```

The reproduction quality of TDXSound is set.

Programming Tips

The reproduction quality of the TDXSound component is set.
It is set to 44100Hz 16 bit stereo here that it is default and 22050Hz 8 bit stereo.
soExclusive of the Options property should be True.
The reproduction quality is set here because of the OnInitialize event.

```
uses Wave;

procedure TForm1.DXSound1Initialize(Sender: TObject);
var
  WaveFormat: TWaveFormatEx;
begin
  {  44100Hz, 16bit, Stereo  }
  MakePCMWaveFormatEx(WaveFormat, 44100, 16, 2);
  DXSound1.Primary.SetFormat(WaveFormat);
end;
```

Template

Template Lenqend

In Template

Lenqend

Scope

protected
published

Accessibility

Read-only
Write-only