TCamRemote Delphi component version 1.2.

TCamRemote

The Delphi component to handle PowerShot/EOS digital cameras

---

by Hans-David Alkenius

You have a Canon digital camera? Want to use the camera as a picture source to your computer? Maybe want to create a movie?

The TCamRemote component for Delphi enables you to handle PowerShot and/or EOS digital cameras within your development platform.

TCamRemote

Printed: maj 2005 in Linköping, Sweden.

Publisher

Hans-David Alkenius

Reviewer

Anders Alkenius
Anna Alkenius

Special thanks to:

My patient wife, which allows me to work with this product. It has been very time consuming and many late hours. Thanks!

Table of Contents

TCamRemote

Part I

1 Introduction

1.1 Welcome

Welcome to the TCamRemote component.

The TCamRemote component can be used to interface and remotely handle Canon PowerShot and EOS digital cameras.

Highlights

- take pictures remotely and receive the picture to the computer,
- handle the remote viewfinder,
- set and get remote parameters (e.g. ISO and Zoom),
- list, get and delete pictures stored in the camera.

1.2 Description

Authors
Hans-David Alkenius, tcamremote@alkenius.no-ip.org

Description

TCamRemote is a Delphi components used to handle Canon PowerShot and/or EOS digital cameras.

Updates

The homesite for the TCamRemote Component http://alkenius.no-ip.org/TCamRemote/

Updates are also posted to Torry's Delphi Pages and Delphi Super Page from time to time.

Delphi requirements
The registered version compiles on any version of Delphi 5, 6, 7 or 2005. More Delphi platforms can probably handle TCamRemote but it has not been tested.
The evaluation version includes object code and is therefore dependent of updated packs installed. The table below lists the version of Delphi including updated packs that the evaluation version was compiled with:

Delphi version	Update packs	Version as according to the about box
Delphi 5	Update pack 1	Version 5.0 (Build 6.18) Update Pack 1
Delphi 6	Update pack 2	Version 6.0 (Build 6.240) Update Pack 2
Delphi 7	Update pack 1	Version 7.0 (Build 8.1)
Delphi 2005	Update pack 2	Version 9.0.1882.30496 Update 2

"[Fatal Error] CamRemote.pas(10): Unit CDSDKApi **was compiled with a different version of** CamDefines.ImageListType" is a typical error message if the Delphi version (including update packs) not is correct for the evaluation version.

Development Requirements
IBM PC/AT, PS/2 or compatible PC:
Host computer
Minimum configuration:
Pentium or higher processor
At least 64 MB RAM (except Windows XP), at least 128 MB RAM (Windows XP)
800 x 600 pixel, 256 color (8 bit) or higher video adapter and monitor
USB interface or IEEE1394 interface
Recommended configuration:
300 MHz or higher Pentium processor

At least 128 MB RAM (except Windows XP), at least 256 MB RAM (Windows XP)
1024 x 768 pixel, High Color (16 bit) or higher video adapter and monitor
USB interface or IEEE1394 interface
Operating System
Windows 98, Windows Me, Windows 2000, Windows XP
TCamRemote supports Delphi 5, 6, 7 & 2005.

Target Requirements
Not only must the camera be supported by this version of TCamRemote, a target system that meets the following requirements is necessary to run the client application created.
IBM PC/AT, PS/2 or compatible PC:
Host computer
Minimum configuration:
Pentium or higher processor
At least 64 MB RAM (except Windows XP), at least 128 MB RAM (Windows XP)
800 x 600 pixel, 256 color (8 bit) or higher video adapter and monitor
Recommended configuration:
Pentium or higher processor
At least 128 MB RAM (except Windows XP), at least 256 MB RAM (Windows XP)
1024 x 768 pixel, True Color (24 bit) or higher video adapter and monitor
Operating System
Windows98, Windows Me, Windows 2000, Windows XP

Suggestions for Improvements

The author welcomes suggestions for improvements and new functions. As long as a function not is requested, it is not implemented in TCamRemote.
Examples of functions that could be implemented are:

- Handle pictures stored on local drive on computer.
- Handle movies and sounds.
- RAW development.
- Upload pictures to the camera.

How it happened...

From the beginning (year 2000) I developed a tool called Cam4you utilities. Cam4you utilities homepage is http://jpegclub.org/cam4you. I used Delphi to develop Cam4you utilities and I decided in 2004 to develop a Delphi VCL making it possible for other Delphi developers to handle Canon cameras.

1.3 License

As a proprietary product, TCamRemote is protected by copyright laws. At all times Hans-David Alkenius retains full title to the software. Subject to your acceptance of and accordance with the terms and conditions stated in this agreement, you shall be granted a single-user software license. Any previous agreement with Hans-David Alkenius is superseded by this agreement.

THIS SOFTWARE LICENSE GIVES YOU THE RIGHT TO:

1. Install and use the Software for the sole purposes of designing, developing, testing, and deploying application programs which you create. You may install a copy of the Software on a computer and freely move the Software from one computer to another, provided that you are the only individual using the Software. If you are an entity, you must designate one individual within your organization ("Named User") to have the right to use the Software.
2. Write and compile your own application programs using the TCamRemote software contained in this package. All copies of the software you so write and distribute must include a TCamRemote copyright notice, or a valid copyright notice of your own.
3. Make one copy of the Software for backup or archival purposes or copy the Software to a single permanent storage medium provided you keep the original solely for backup or archival purposes.
4. Distribute runtime packages for the sole purpose of executing application programs created with Delphi. TCamRemote runtime packages available for distribution are listed in the

TCamRemote runtime files chapter.

ENGAGING IN ANY OF THE ACTIVITIES LISTED BELOW WILL TERMINATE THE SOFTWARE LICENSE.

1. Distribution of any files contained in this software package, other than the runtime packages explicitly listed above, including but not limited to .PAS, .DFM, .DCU files, .DCP files, and design-time packages.
2. Modification, de-compilation, disassembly, reverse engineering or translation of the Software.
3. Removal of proprietary notices, labels or marks from the Software or Software Documentation.
4. Inclusion of the software in a development environment.
5. Creation of an application that does not differ materially from the Software.
6. Creation of an application (whether it will be freeware, shareware or a commercial product) which competes directly or indirectly with TCamRemote.
7. Distribution of an application program created using the Software to another developer. A developer is defined as any person who is executing an application program created using the Software, on a computer which contains an installation of Borland Delphi. In order to execute such an application, the developer must own a license to the Software, and must have installed the Software on the computer.
8. You may not use TCamRemote to create components or controls to be used by other developers without written approval from Hans-David Alkenius.

AGREEMENT PERTAINING TO THE RELEASE OF SOURCE CODE by Hans-David Alkenius to Recipient:

USE OF SOURCE CODE

Recipient will not utilize the source for the creation of software (whether it is freeware, shareware or a commercial product) which competes directly or indirectly with TCamRemote. In addition, Recipient will not disclose the source itself, nor the implementations discovered therein, to any party involved in the creation of software which competes directly or indirectly with TCamRemote.

DISTRIBUTION OF SOURCE CODE

Recipient will not distribute the source. Specifically this includes all .dcu, .dfm, and .pas files which Hans-David Alkenius has provided.

CHANGES TO SOURCE CODE

Hans-David Alkenius reserves the right to change any part of the source in future versions of the product. These changes may include the removal of classes, properties and methods or the creation of new classes, properties and methods.

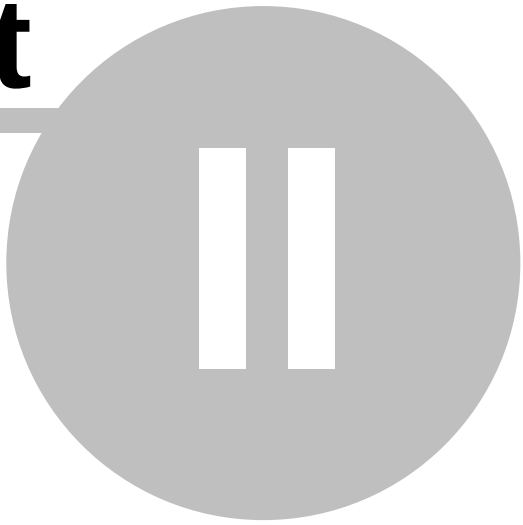
TECHNICAL SUPPORT FOR SOURCE CODE

The Software is not guaranteed to be error free. Hans-David Alkenius will provide limited technical support but will not provide support for changes recipient makes to the source. Recipient assumes full responsibility for supporting any code or application which results from such modification. Recipient will not hold Hans-David Alkenius liable, directly or indirectly, for any changes made to the source, including changes which Recipient has made based on advice or suggestions provided by Hans-David Alkenius. The Recipent shall get back the payment for the Software, if critical errors found in the Software not can be solved by Hans-David Alkenius. Recipent will not hold Hans-David Alkenius liable for any hardware problems when using the Software.

1.4 Evaluation of TCamRemote

TCamRemote is available as a registered version and as an evaluation version. The registered version consists of all source code implementing TCamRemote, including all documentation and template applications. The evaluation copy consists of precompiled objects of TCamRemote that can be used to test the TCamRemote functions. No function is disabled in the evaluation version, but a message box will be displayed when an application using TCamRemote is executed outside the Delphi IDE.

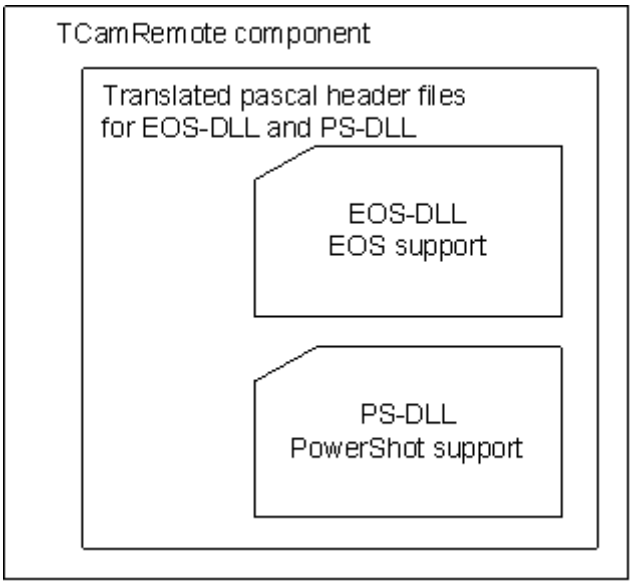
Part



2 Overview

2.1 Overview

The TCamRemote can support PowerShot and/or EOS digital cameras. The support is defined as compiler switches defined in the CamRemote.inc files.
Depending on which cameras TCamRemote supports, required run-time DLLs must be reached by the application using TCamRemote.



In summary, the TCamRemote component contains the following:

File	Use
CamRemote.pas	The TCamRemote component interface unit.
CamDefines.pas	Types and classes used by the TCamRemote component.
CamRemote.inc	Compiler flags which sets support for PowerShot cameras (PS-DLL) or EOS cameras (EOS-DLL).
CamLib.pas	Library function used by TCamRemote.
CDSDKApi.pas	The interface unit to PS-DLL.
RCSDKApi.pas	The interface unit to EOS-DLL.
CDAPI.PAS	Translation of PS-DLL header files.
CDERRORCODES.PAS	Translation of PS-DLL and EOS-DLL header files.
CDEVENT.PAS	Translation of PS-DLL and EOS-DLL header files.
CDTYPE.PAS	Translation of PS-DLL header files.
RCAPI.pas	Translation of EOS-DLL header files.
RCTYPE.PAS	Translation of EOS-DLL header files.

2.2 Supported cameras

The following camera are supported:

PS-DLL:
PowerShot A10, PowerShot A20, PowerShot A30, PowerShot A40, PowerShot A60, PowerShot A70, PowerShot A75, PowerShot A80, PowerShot A85, PowerShot A95

PowerShot A100, PowerShot A200, PowerShot A300, PowerShot A310, PowerShot A400
PowerShot S10, PowerShot S20, PowerShot S30, PowerShot S40, PowerShot S45, PowerShot S50, PowerShot S60, PowerShot S70
PowerShot S100, IXY DIGITAL, DIGITAL IXUS
PowerShot S110, IXY DIGITAL 200, DIGITAL IXUS v
PowerShot S200, IXY DIGITAL 200a, DIGITAL IXUS v2
PowerShot S230, IXY DIGITAL 320, DIGITAL IXUS v3
PowerShot S300, IXY DIGITAL 300, DIGITAL IXUS 300
PowerShot S330, IXY DIGITAL 300a, DIGITAL IXUS 330
PowerShot SD100, IXY DIGITAL 30, DIGITAL IXUS II
PowerShot S400, IXY DIGITAL 400, DIGITAL IXUS 400
PowerShot SD10, IXY DIGITAL L, DIGITAL IXUS I
PowertShot SD110, IXY DIGITAL 30a, DIGITAL IXUS IIs
PowerShot S410, IXY DIGITAL 450, DIGITAL IXUS 430
PowerShot S500, IXY DIGITAL 500, DIGITAL IXUS 500
PowerShot SD20, IXY DIGITAL L2 , DIGITAL IXUS i5
PowerShot SD200, IXY DIGITAL 40, DIGITAL IXUS 30
PowerShot SD300, IXY DIGITAL 50, DIGITAL IXUS 40
PowerShot G1, PowerShot G2, PowerShot G3, PowerShot G5, PowerShot G6
PowerShot Pro90 IS
PowerShot S1 IS, PowerShot Pro1

The PS-DLL has also been designed to be compatible with future Canon digital camera models. However, the PS-DLL does not support connections to a camera using an RS-232C interface. As a result, the camera models shown below cannot be accessed from an application that uses the PS-DLL through an RS-232C interface. If you are using one of these camera models, you must connect it to the computer using a USB interface to access the camera from the PS-DLL application.
PowerShot G1, PowerShot Pro90 IS

EOS-DLL:
EOS D30, D60, 10D, 20D, Kiss Digital, Digital Rebel, 300D
EOS-1D, 1Ds, 1Ds MarkII

2.3 Revision History

Version 1.2
- Changes in the documentation.

Version 1.1
- The VCL evaluation version is compiled using the latest update packs for Delphi 5, 6 and 7.
- The VCL evaluation version for Delphi 2005 does not display any nag screen while the IDE is running.

Version 1.0
- First version of TCamRemote.

2.4 Installation

Adding the components to the Component Palette:

1. If you install the registered version, set the compiler switches in CamRemote.inc as described below.

2. In the Delphi IDE select "File|Open" from the menu.

3. Open the appropriate component package file from the folder where you installed the components. The package file is named:

    * TCamRemoteD5.dpk for Delphi 5
    * TCamRemoteD6.dpk for Delphi 6
    * TCamRemoteD7.dpk for Delphi 7

4. In the Package Dialog click the Install button. TCamRemote will be installed in the Samples tab in Delphi IDE.

5. Click the Options button and then the Global tab.
   Add the path of the component units to the Library Path.

Adding the TCamRemote.hlp file to Delphi's help system:

1. In the IDE select Help|Customize ... from the menu.

2. When the OpenHelp dialog appears select the Index tab.

3. Click the Add File toolbar button and add TCamRemote.hlp which
   will be found in the folder where you saved the components.

4. Close the OpenHelp dialog and Save when prompted.

5. It may be necessary to close Delphi and
   restart before the help system is properly updated.

Use Help|Delphi help and then Index to search for TCamRemote and its methods/types etc. It is possible that this enables the context sensivity help in Delphi 5, but for Delphi 6 and 7 no context sensitivity help is available for TCamRemote.

Compiler switches used in the registered version of TCamRemote:
The CamRemote.inc file in the Sources directory includes compiler switches that enables/disables functions in TCamRemote. Before installing enable/sidable these compiler swicthes.

To remove a compiler switch set a "." before the switch name.
{$DEFINE PSDLL} //The PSDLL switch is enabled
{.$DEFINE PSDLL} //The PSDLL switch is disabled

Switch	Description
PSDLL	Enables support for PowerShot cameras. When enabled it requires that the runtime files for PS-DLL is copied to application directory.
EOSDLL	Enables support for EOS cameras. When enabled it requires that the runtime files for PS-DLL is copied to application directory.
DEXIF	Enables support for EXIF in TCamRemote. It requires that the dEXIF component is installed, see below.
TCAMREMOTE_REGISTRED	Shall always be enabled.

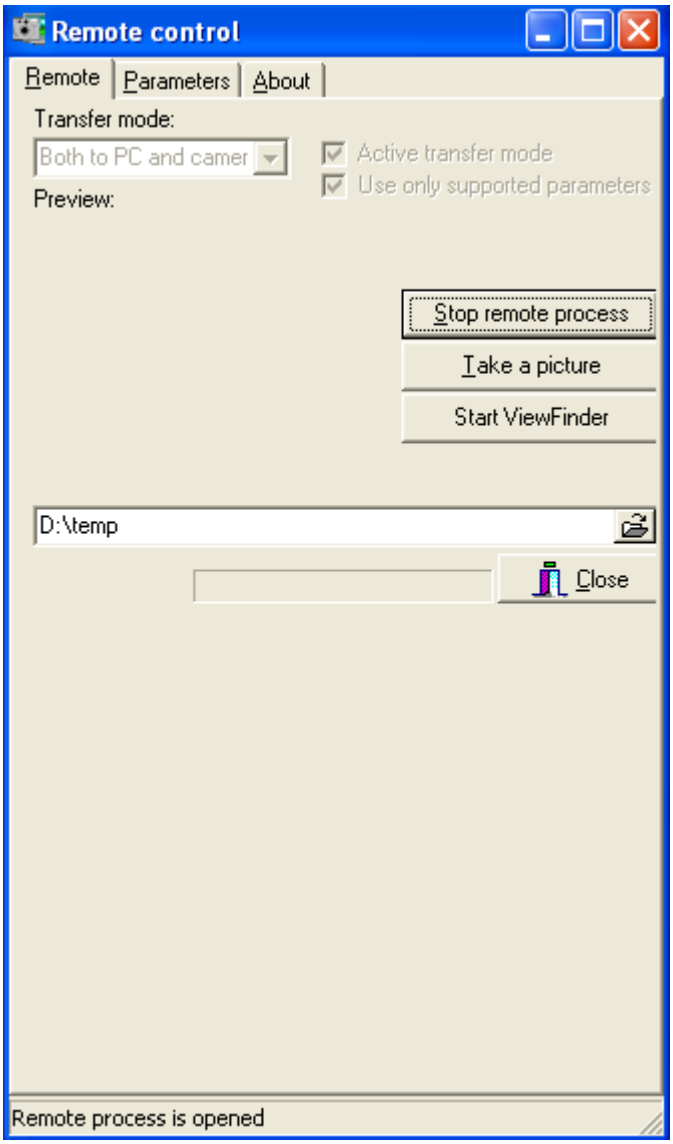
Install dEXIF to add EXIF support:
TCamRemote can extract EXIF information from RAW-pictures, using the dEXIF component created by Gerry McGuire. The source code including installation instruction is stored in the /Sources/dEXIF directory. This component enables TCamRemote to return EXIF information for pictures stored on the camera.

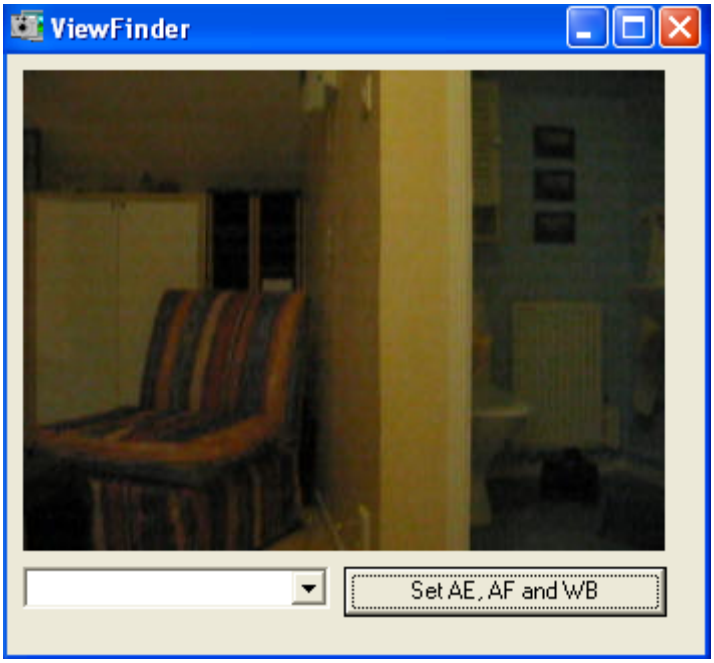
2.5 Template applications

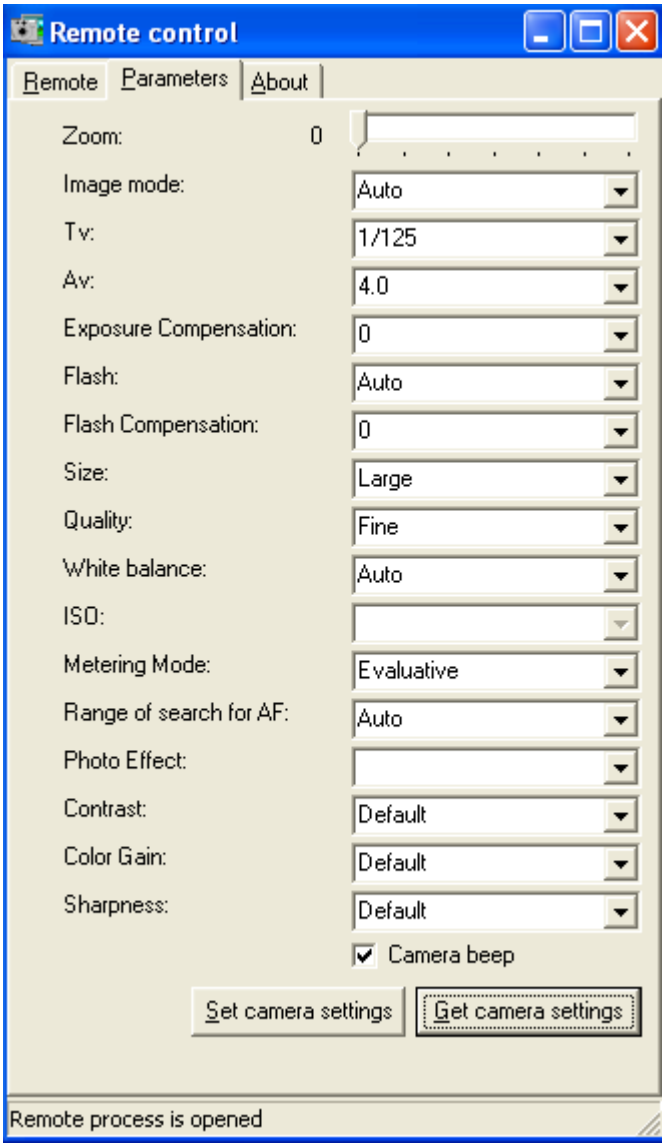
RemoteTemplate:
The RemoteTemplate application demonstrates how the TCamRemote component is used to set and get remote parameters, handles remote viewfinder and take pictures remotely.

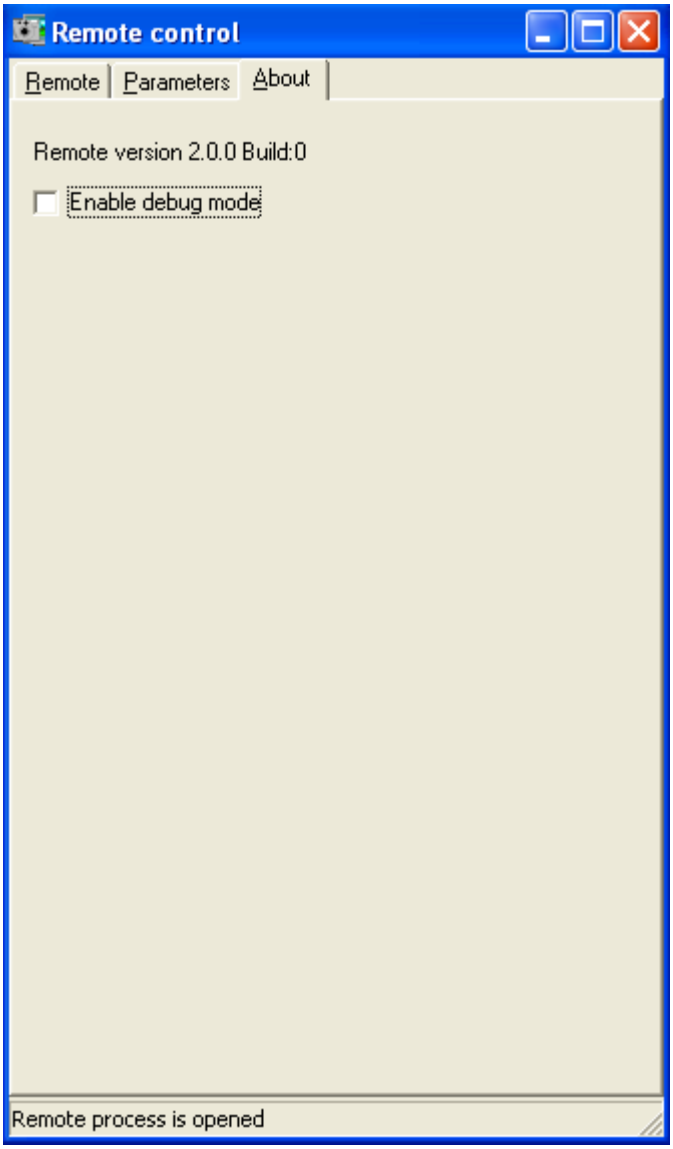
When the checkbox "Active transfer mode" is checked, active transfer mode is used (UseSyncMode is

set to true) when calling the RemoteTakePicture method.
When the checkbox "Use only supported parameters is checked, only supported remote parameters probed when calling the RemoteStart method is showed in the Parameters tab. When not checked all remote parameters are showed in the Parameters tab.

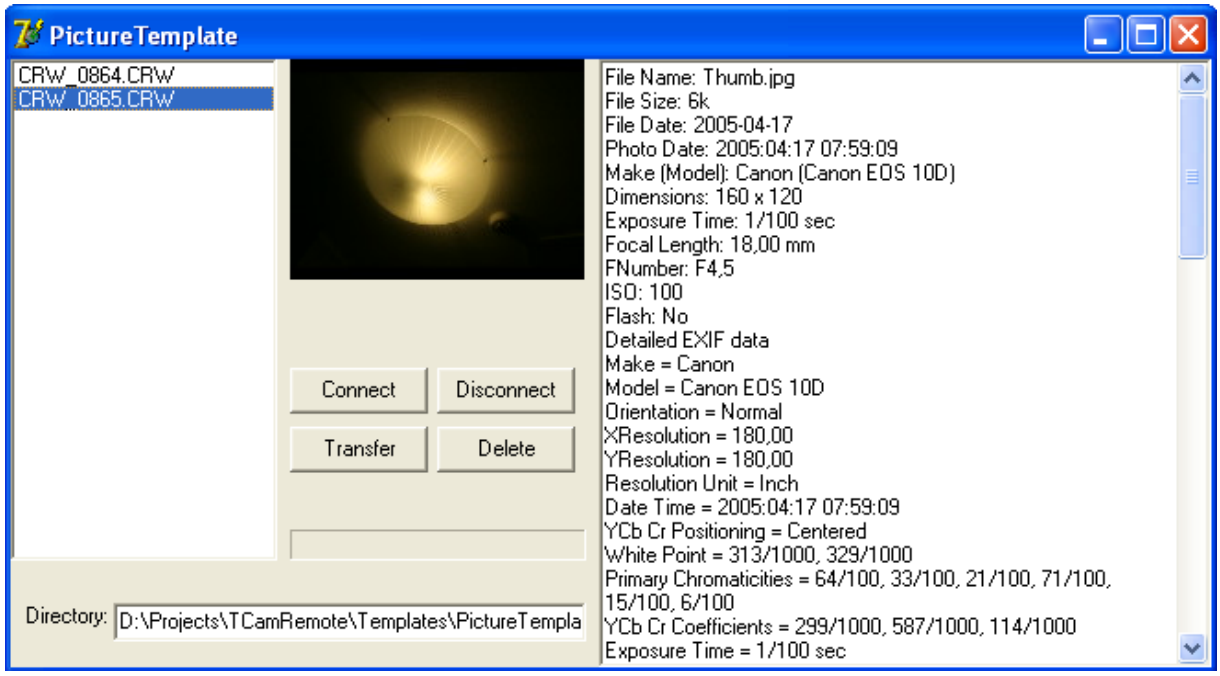




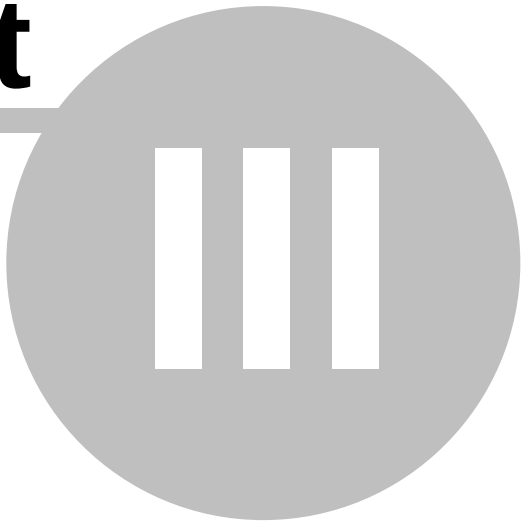




PictureTemplate:
The PictureTemplate application lists pictures stored on the camera when the Connect button is pressed. EXIF information is displayed for selected RAW pictures (requires that dEXIF is enabled during installation) and that the DEXIF compiler switch is enabled in the PictureTemplate.inc include file. Selected picture can be transferred to the computer pressing the Transfer button and deleted pressing the Delete button.



Part



3 TCamRemote component

3.1 TCamRemote

TCamRemote enables applications to remotely handle a Canon PowerShot or EOS digital camera.

Unit
CamRemote

Declaration
```
TCamRemote = class(TComponent)
```

Description
Use TCamRemote to connect to a PowerShot or EOS digital camera. With TCamRemote it is possible to

- handle the remote viewfinder,
- set and get remote parameters (e.g. ISO),
- take pictures remotely and receive the picture to the computer,
- list, get and delete pictures stored in the camera.

3.2 TCamRemote runtime files

The CamRemote uses DLLs to interface the PowerShot and EOS cameras. The DLLs are stored in the Redist directory.

- PS-DLL are used to interface PowerShot cameras.
- EOS-DLL are used to interface EOS cameras.

CamRemote must reach the required DLLs to operate correctly, which requires that the DLLs either are copied to the application directory or that the DLLs can be reached using the PATH environment variable. It is recommended to copy the DLLs to the application directory.

3.3 TCamRemote error handling

TCamRemote uses exceptions when errors occurs (e.g. when a request for connection to a camera fails). TCamRemote uses the ECamException class defined below:

Unit
CamDefines

Syntax:
```
ECamException = class(Exception);
```

The exception may include an error code and a textual interpretation of the error. The interpretation of error codes are listed here.

3.4 TCamRemote methods

3.4.1 List of methods in TCamRemote

RemoteEnd
RemoteGetPicture
RemoteGetRemoteParams
RemoteGetZoomPos
RemoteSetRemoteParams
RemoteSetViewfinderOutput
RemoteSetZoomPos
RemoteStart
RemoteStartViewfinder
RemoteStopViewfinder
RemoteSupported
RemoteTakePicture

3.4.2 TCamRemote.DeletePicture

Deletes a picture in the camera.

Unit
CamRemote

Syntax:
```
procedure DeletePicture(var ImageList : ImageListType; NrInList : integer);
```

Prerequisite:
A successful connection must have been established with the Connect method.
A successful call to the OpenCameraCollection method, to list pictures on the camera.

Description
The DeletePicture method deletes a picture in the camera.

Please note the following:
- The ImageList parameter is modified by this method. Please reparse the array after a call to DeletePicture method has been performed.

Parameter:
- ImageList: The data structure returned from OpenCameraCollection.
- NrInList: The array element in ImageList that is to be deleted in the camera.

If any errors occurs an ECamException exception will be raised.

3.4.2.1 DeletePicture example

```
procedure TFormMain.ButtonDeleteClick(Sender: TObject);
begin
  if (ListBox.ItemIndex <> -1) then
  begin
    CamRemote.DeletePicture(mImageList, ListBox.ItemIndex);
    //Update the list of pictures, since mImageList has been modified
    UpdateListBox;
  end else begin
    MessageDlg('No picture selected to delete', mtInformation, [mbOK], 0)
  end;
end;

procedure TFormMain.UpdateListBox;
var i : integer;
begin
  //Update the list of pictures on the form
  ListBox.Items.Clear;
  for i := 0 to Length(mImageList) - 1 do
  begin
    ListBox.Items.Add(mImageList[i].Name)
  end;
end;
```

3.4.3 TCamRemote.GetPicture

Copies a picture from the camera to a file on the computer.

Unit
CamRemote

Syntax:
```
procedure GetPicture(NrInList : integer; FileName : string);
```

Prerequisite:
A successful connection must have been established with the Connect method.
A successful call to the OpenCameraCollection method, to list pictures on the camera.

Description
The GetPicture method copies the picture selected by the NrInList parameter to the file specified in the FileName parameter.
The OnGetPictureEvent is used during transfer.

Parameter:
- NrInList: The array element in ImageList (received from method OpenCameraCollection) that is to be copied to the computer.
- FileName: The filename for the copied file on the computer.

If any errors occurs an ECamException exception will be raised.

3.4.3.1 GetPicture example

```
//The user has pressed the transfer button
procedure TFormMain.ButtonTransferClick(Sender: TObject);
begin
  if (ListBox.ItemIndex <> -1) then
  begin
    CamRemote.GetPicture(ListBox.ItemIndex,
                         EditPCDir.Text + '\' + ListBox.Items[ListBox.ItemIndex]);
  end;
end;

//GetPictureEvent callback used during filetransfer to computer
procedure TFormMain.CamRemoteGetPictureEvent(PercentageDone: Integer);
begin
  ProgressBar.Position := PercentageDone;
end;
```

3.4.4 TCamRemote.CloseCameraCollection

Closes the volume for pictures on the camera and frees the data structure returned from the OpenCameraCollection method.

Unit
CamRemote

Syntax:
```
procedure CloseCameraCollection(var ImageListToClear : ImageListType);
```

Prerequisite:
A successful connection must have been established with the Connect method.
A successful call to the OpenCameraCollection method, to list pictures on the camera.

Description
The CloseCameraCollection method closes the volume on the camera, which is used to handle pictures in the camera. The data structure returned from OpenCameraCollection (supplied as parameter to the method) is freed.

Parameter:

- ImageListToClear: The data structure returned from OpenCameraCollection.

If any errors occurs an ECamException exception will be raised.

3.4.4.1 CloseCameraCollection example

```
//The user has pressed the disconnect button
procedure TFormMain.ButtonDisconnectClick(Sender: TObject);
begin
  //Close the picture collection and delete the objects in mImageList
  CamRemote.CloseCameraCollection(mImageList);
  CamRemote.Disconnect;
end;
```

3.4.5 TCamRemote.Connect

Connects to the camera and returns information of the camera connected.

Unit
CamRemote

Syntax:
```
function Connect: ConnectInfoType;
```

Prerequisite:
The runtime files needs to be copied to the directory from which the application is executing.
The camera needs to be connected to the computer and the connection must be active.

Description
The Connect method tries to establish a connection to a connected camera. Only one camera can be connected at the same time and if a PowerShot as well as an EOS camera are connected to the computer, the PowerShot camera will be connected using the Connect method. The connection to the camera will remain until the Disconnect method is called or the camera is disconnected from the computer (the USB cable is ejected or batterry level very critical).
After a successful connection

- information about the connected camera is returned,
- the OnEvent event is called, when a camera event has occurred.

If any errors occurs an ECamException exception will be raised.

3.4.6 TCamRemote.Create

Creates a new TCamRemote object.

Unit
CamRemote

Syntax:
```
constructor Create (AOwner: TComponent); override;
```

Description
The constructor for a TCamRemote object.

Parameter:
- AOwner: Sets owner of the TCamRemote object.

If any errors occurs an ECamException exception will be raised.

3.4.7 TCamRemote.Destroy

Destroys the TCamRemote instance and frees its memory.

Unit
CamRemote

Syntax:
```
destructor Destroy; override;
```

Description
Do not call Destroy directly. Call Free instead. Free checks that the object reference is not nil before calling Destroy.

3.4.8 TCamRemote.Disconnect

Disconnects from the connected camera.

Unit
CamRemote

Syntax:
```
procedure Disconnect;
```

Prerequisite:
A successful connection must have been established with the Connect method.

Description
The Disconnect method disconnects the camera from the TCamRemote object. When disconnected no more events from the camera will occur, therefore the OnEvent event will not be called anymore.

If any errors occurs an ECamException exception will be raised.

3.4.9 TCamRemote.GetOwnerName

Returns the owner name stored in the camera.

Unit
CamRemote

Syntax:
```
function GetOwnerName: string;
```

Prerequisite:
A successful connection must have been established with the Connect method.

Description
An owner name can be stored in the camera. This method returns the owner name.

If any errors occurs an ECamException exception will be raised.

3.4.10 TCamRemote.OpenCameraCollection

Returns a list of objects that represents meta data for pictures stored on the camera.

Unit
CamRemote

Syntax:
```
function OpenCameraCollection(TempDir : string) : ImageListType;
```

Prerequisite:
A successful connection must have been established with the Connect method.

Description
OpenCameraCollection opens a volume on the camera and searches for pictures stored on the camera. Pictures meta data (e.g. name, thumbnail and EXIF) is returned.

Please note the following:
- The user of the OpenCameraCollection method is responsible to save and free the data structure returned from this method. This can be achieved by calling the CloseCameraCollection method and pass the data structure as parameter.

Parameter:
- TempDir: A temporary directory where TCamRemote can create temporary thumbnail files, when pictures are searched in the camera.

If any errors occurs an ECamException exception will be raised.

3.4.10.1 OpenCameraCollection example

```
procedure TFormMain.ButtonConnectClick(Sender: TObject);
begin
   //Connect to the camera
   CamRemote.Connect;
   //List all pictures on the camera
   mImageList := CamRemote.OpenCameraCollection(ExtractFileDir(ParamStr(0)));
   //Update the list of pictures on the form
   ListBox.Items.Clear;
   for i := 0 to Length(mImageList) - 1 do
   begin
     ListBox.Items.Add(mImageList[i].Name)
   end;
end;

//Procedure called when user clicks on a picture filename on the ListBox updated above.
procedure TFormMain.ListBoxClick(Sender: TObject);
{$IFDEF DEXIF}
var exif_item : TTagEntry;
{$ENDIF}
begin
  //Print EXIF data for RAW-pictures to a RichEdit VCL.
  RichEdit.Lines.Clear;
  if (ListBox.ItemIndex <> -1) then
  begin
    //Update thumbnail on the form
    ImageThumbnail.Picture.Assign(mImageList[ListBox.ItemIndex].Thumbnail);
{$IFDEF DEXIF}
    if (mImageList[ListBox.ItemIndex].EXIF.ExifObj <> nil) then
    begin
      RichEdit.Lines.Add(mImageList[ListBox.ItemIndex].EXIF.ExifObj.toLongString);
      RichEdit.Lines.Add('Detailed EXIF data');
      mImageList[ListBox.ItemIndex].EXIF.ExifObj.ResetIterator;
      while mImageList[ListBox.ItemIndex].EXIF.ExifObj.IterateFoundTags(GenericEXIF,
exif_item) do
      begin
        RichEdit.Lines.Add(exif_item.Desc+DexifDelim + exif_item.Data);
      end;
      RichEdit.Lines.Add('Maker specific EXIF data');
      mImageList[ListBox.ItemIndex].EXIF.ExifObj.ResetIterator;
      while mImageList[ListBox.ItemIndex].EXIF.ExifObj.IterateFoundTags(CustomEXIF, exif_item)
do
      begin
        RichEdit.Lines.Add(exif_item.Desc + DexifDelim + exif_item.Data);
      end;
    end;
{$ENDIF}
  end;
end;
```

3.4.11 TCamRemote.RemoteActivateViewfinderAuto

Forces the camera to re-execute AE (Auto exposure) and AF (Autofocus) for remote viewfinder.

Unit
CamRemote

Syntax:
```
procedure RemoteActivateViewfinderAuto;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
The remote viewfinder must be active, set by the RemoteStartViewfinder method.

Description
When light or target condition changes this method is used to force the camera to recalculate remote viewfinder AE and AF.

Please note the following:
- EOS cameras does not support remote viewfinder.

If any errors occurs an ECamException exception will be raised.

3.4.12 TCamRemote.RemoteEnd

Ends remote mode.

Unit
CamRemote

Syntax:
```
procedure RemoteEnd;
```

Prerequisite:
The camera must successfully have been set into remote mode using the RemoteStart method.

Description
The remote mode will be closed. The camera lens will be withdrawn, if applicable. The connection to the camera (started by calling the Connect method) will still remain. The OnRemoteEvent event will not be called anymore.

If any errors occurs an ECamException exception will be raised.

3.4.13 TCamRemote.RemoteGetPicture

Gets a remotely taken picture.

Unit
CamRemote

Syntax:
```
procedure RemoteGetPicture (FileName: string);
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
A picture has remotely been taken, using a call to the RemoteTakePicture method.

Description
RemoteGetPicture copies the picture data to a file with a filename set by the FileName parameter. If a thumbnail as well as a picture are requested (set by the parameter ReleaseDataKind in the RemoteStart method), the thumbnail is received only when calling RemoteGetPicture. The full picture

is then received when calling the RemoteGetPicture a second time.
The OnRemoteGetPictureEvent is used during transfer.

Please note the following:

- It is not possible to receive any picture data if the ReleaseDataKind parameter in the RemoteStart method is set to ReleaseModeOnlyToCamera.

Parameter:
- FileName: The filename for the picture to receive.

If any errors occurs an ECamException exception will be raised.

3.4.13.1 RemoteGetPicture example

```
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic);
  //Take a picture. Sync = true
  num_pics := CamRemote.RemoteTakePicture(true);
  if (num_pics = 2) then
  begin
    //First get the thumbnail
    CamRemote.RemoteGetPicture('C:\Temp\Thumb.jpg');
    //Then get the picture
    CamRemote.RemoteGetPicture('C:\Temp\Pic.jpg');
  end;
```

3.4.14 TCamRemote.RemoteGetRemoteParams

Gets the current used remote parameters from the camera.

Unit
CamRemote

Syntax:
```
function RemoteGetRemoteParams: RemoteReleaseParametersType;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
Check DoSupportShootingPara in RemoteFuncType returned by the Connect method, before calling this method.

Description
-

If any errors occurs an ECamException exception will be raised.

3.4.14.1 RemoteGetRemoteParams example

```
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic);
  //Check if remote parameters are supported
  if (not mCameraCapability.DoSupportShootingPara) then
  begin
    ECamException.Create('Shooting parameters not supported by the connected camera');
  end;
  //Check if it is possible to set Av to 1/60 sec.
  //Check if that parameter is supported by the camera
  if (not mCameraCapability.RemoteParamSupported.Tv[RemoteFormatTV1_60]) then
  begin
    if (MessageDlg('The 1/60 Tv value not supported. ' +
                    'Do you want to try to set that value event that it may not be
supported?',
                   mtConfirmation, [mbOK, mbCancel], 0) = mrCancel) then
    begin
      exit;
    end;
```

```
end;

//Get the current remote parameters
remote_parameters := CamRemote.RemoteGetRemoteParams;
remote_parameters.Tv := RemoteFormatTV1_60;
//Set the remote parameters
CamRemote.RemoteSetRemoteParams(remote_parameters);
```

3.4.15 TCamRemote.RemoteGetZoomPos

Gets camera zoom position and zoom capabilities.

Unit
CamRemote

Syntax:
```
function RemoteGetZoomPos: RemoteZoomCapabilityType;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
Check DoSupportZoom in RemoteFuncType returned by the Connect method, before calling this method.

Description
-

Please note the following:
- EOS cameras does not support zoom. Instead zoom is changed manually on the camera lens.

If any errors occurs an ECamException exception will be raised.

3.4.15.1 RemoteGetZoomPos example

```
//Check if zoom is supported
if (not mCameraCapability.DoSupportZoom) then
begin
  ECamException.Create('The connected camera does not support zoom');
end;
//Get the zoom position
zoom_parameters := CamRemote.GetZoomPos;
//Set the zoom to max optical zoom
zoom_parameters.CurrentZoomPos := zoom_parameters.MaxOpticalZoomPos;
CamRemote.SetZoomPos(zoom_parameters);
//Set the zoom to max digital zoom
zoom_parameters.CurrentZoomPos := zoom_parameters.MaxZoomPos;
CamRemote.SetZoomPos(zoom_parameters);
```

3.4.16 TCamRemote.RemoteSetRemoteParams

Sets remote parameters to the camera.

Unit
CamRemote

Syntax:
```
procedure RemoteSetRemoteParams (RemoteParam: RemoteReleaseParametersType);
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
Check DoSupportShootingPara in RemoteFuncType returned by the Connect method, before calling this method.

Description
Supported remote parameters from the camera are listed in RemoteParamSupported returned from the RemoteStart method. These parameters have been verified for camera acceptence and are the

only recommended. However it is allowed to set any remote parameters to the camera, but do not expect that the camera accepts them. Use the RemoteGetRemoteParams method to verify which parameters that are accepted or not.

Parameter:
- RemoteParam: The remote parameters set to the camera.

If any errors occurs an ECamException exception will be raised.

3.4.16.1 RemoteSetRemoteParams example

```
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic);
  //Check if remote parameters are supported
  if (not mCameraCapability.DoSupportShootingPara) then
  begin
    ECamException.Create('Shooting parameters not supported by the connected camera');
  end;
  //Check if it is possible to set Av to 1/60 sec.
  //Check if that parameter is supported by the camera
  if (not mCameraCapability.RemoteParamSupported.Tv[RemoteFormatTV1_60]) then
  begin
    if (MessageDlg('The 1/60 Tv value not supported. ' +
                    'Do you want to try to set that value event that it may not be
supported?',
                   mtConfirmation, [mbOK, mbCancel], 0) = mrCancel) then
    begin
      exit;
    end;
  end;

  //Get the current remote parameters
  remote_parameters := CamRemote.RemoteGetRemoteParams;
  remote_parameters.Tv := RemoteFormatTV1_60;
  //Set the remote parameters
  CamRemote.RemoteSetRemoteParams(remote_parameters);
```

3.4.17 TCamRemote.RemoteSetViewfinderOutput

Changes the remote viewfinder output destination.

Unit
CamRemote, CamDefines

Syntax:
```
procedure RemoteSetViewfinderOutput (ViewfinderOutput: RemoteViewFinderOutputType);

type RemoteViewFinderOutputType = (RemoteViewFinderOutputLCD,
                                   RemoteViewFinderOutputVideo,
                                   RemoteViewFinderOutputOff);
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
The remote viewfinder must be active, set by the RemoteStartViewfinder.

Description
The remote viewfinder destination can be changed by calling this method.

Please note the following:
- EOS cameras does not support remote viewfinder.

Parameter:
- ViewfinderOutput: The destination for pictures taken by the remote viewfinder..

If any errors occurs an ECamException exception will be raised.

3.4.18 TCamRemote.RemoteSetZoomPos

Sets camera zoom position.

Unit
CamRemote

Syntax:
```
procedure RemoteSetZoomPos (ZoomPos: integer);
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
Check DoSupportZoom in RemoteFuncType returned by the Connect method, before calling this method.

Description
-

Please note the following:
- EOS cameras does not support zoom. Instead zoom is changed manually on the camera lens.

Parameter:
- ZoomPos: The requested zoom position. ZoomPos must be lower than MaxZoomPos in RemoteZoomCapabilityType returned from the RemoteGetZoomPos method.

If any errors occurs an ECamException exception will be raised.

3.4.18.1 RemoteSetZoomPos example

```
//Check if zoom is supported
if (not mCameraCapability.DoSupportZoom) then
begin
  ECamException.Create('The connected camera does not support zoom');
end;
//Get the zoom position
zoom_parameters := CamRemote.GetZoomPos;
//Set the zoom to max optical zoom
zoom_parameters.CurrentZoomPos := zoom_parameters.MaxOpticalZoomPos;
CamRemote.SetZoomPos(zoom_parameters);
//Set the zoom to max digital zoom
zoom_parameters.CurrentZoomPos := zoom_parameters.MaxZoomPos;
CamRemote.SetZoomPos(zoom_parameters);
```

3.4.19 TCamRemote.RemoteStart

Starts the camera remote mode and returns remote capability parameters for the connected camera.

Unit
CamRemote, CamDefines

Syntax:
```
function RemoteStart(ReleaseMode: ReleaseModeType; ReleaseDataKind: ReleaseDataKindType):
RemoteFuncType;

type ReleaseModeType =  (ReleaseModeOnlyToPC,
                         ReleaseModeBothPCAndCamera,
                         ReleaseModeOnlyToCamera);

     ReleaseDataKindType = (ReleaseDataKindTakeOnlyThumbnail,
                            ReleaseDataKindTakeOnlyPicture,
                            ReleaseDataKingTakeBothThumbAndPic);
```

Prerequisite:
A successful connection must have been established with the Connect method.
Call the RemoteSupported method to check if remote mode is supported by the camera.
The connected camera must be able to extend its lens, if it is withdrawn when the camera is shutting

down.

Description
The camera is put into remote mode, which may cause the lens to be ejected, if it is withdrawn. The camera will also be probed for supported remote parameters (if remote parameters are supported by the camera), which can take some time. This method returns the supported remote capability parameters for the connected camera. The OnRemoteEvent event will be called, when a remote camera event has occurred.

Please note the following:
- The supported remote parameters may differ quite a bit depending on the mode knob on the camera. No shutter or aperture remote parameters can be set if the mode knob on the camera is set to "Auto". They can however be set if the knob is set to "Manual".
- Many remote parameters (e.g. ISO and flash) are not supported by the EOS cameras. These parameters must be set manually on the camera before pictures are taken.
- RemoteParamSupported returned from RemoteStart is only valid if DoSupportShootingPara returned from RemoteStart is true.
- It is not known if all EOS cameras currently are supported. It is unclear whether EOS D20 and cameras created after that camera works with current implementation of TCamRemote. TCamRemote has not been tested with D20 or newer camera yet. Please send feedback to author of TCamRemote if tests are performed.

Parameter:
- ReleaseMode: Sets pictures destination (on the PC and/or camera) used when calling RemoteTakePictures.
- ReleaseDataKind: Sets if thumbnail and/or picture shall be taken when calling RemoteTakePicture.

If any errors occurs an ECamException exception will be raised.

3.4.19.1 RemoteStart example

```
//Check if remote mode is supported
if (not CamRemote.RemoteSupported) then
begin
  ECamException.Create('The connected camera does not support remote mode');
end;
//Updates a status bar
StatusBar.SimpleText := 'Starts remote process and probes remote parameters';
//Starts remote mode
mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                           ReleaseDataKingTakeBothThumbAndPic);
```

3.4.20 TCamRemote.RemoteStartViewfinder

Starts the remote viewfinder.

Unit
CamRemote

Syntax:
```
procedure RemoteStartViewfinder;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
Check DoSupportViewfinder in RemoteFuncType returned by the Connect method, before calling this method.

Description
The camera starts the electronic remote viewfinder. The camera will take 320x240 pictures rapidly and call the OnViewfinderEvent when a picture is available.

Please note the following:

- EOS cameras does not support remote viewfinder.

If any errors occurs an ECamException exception will be raised.

3.4.20.1 RemoteStartViewfinder example

```
//Check if remote viewfinder is supported
if (not mCameraCapability.DoSupportViewfinder) then
begin
  ECamException.Create('The connected camera does not support remote viewfinder');
end;
//Updates a status bar
StatusBar.SimpleText := 'Starts the remote viewfinder';
//Starts remote viewfinder
CamRemote.RemoteStartViewfinder;
```

3.4.21 TCamRemote.RemoteStopViewfinder

Stops the remote viewfinder.

Unit
CamRemote

Syntax:
```
procedure RemoteStopViewfinder;
```

Prerequisite:
The camera remote viewfinder must be active, after calling the RemoteStartViewfinder method.

Description
The remote viewfinder is stopped and no more pictures will be taken in remote mode. The camera will after the remote viewfinder is stopped, still be in remote mode and connected.

Please note the following:
- EOS cameras does not support remote viewfinder.

If any errors occurs an ECamException exception will be raised.

3.4.22 TCamRemote.RemoteSupported

Returns true if the camera supports remote mode.

Unit
CamRemote

Syntax:
```
function RemoteSupported: boolean;
```

Description
Use this function before starting the remote mode, by calling the RemoteStart method.

If any errors occurs an ECamException exception will be raised.

3.4.23 TCamRemote.RemoteTakePicture

Takes a picture and returns the number of pictures that is ready to be read, using the RemoteGetPicture method.

Unit
CamRemote

Syntax:
```
function RemoteTakePicture(UseSyncMode: boolean): integer;
```

Prerequisite:

A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.

Description
Triggers the camera to take a picture remotely. A thumbnail and/or picture will be created in the camera and transferred to the PC depending on the parameters set when calling the RemoteStart method. RemoteTakePicture

- will return when the pictures are available to receive (using the RemoteGetPicture method) or
- will return immediately when the request has been sent to the camera.

RemoteTakePicture will return the number of pictures that is possible to receive.

Please note the following:

- It is possible to take a picture when using remote viewfinder, only if ReqViewfinderOffWhenShooting is false in RemoteFuncType returned by the Connect method.

Parameter:
- UseSyncMode:
SyncMode = true. The RemoteTakePicture method automatically transfers the taken pictures to the computer and saves it within the TCamRemote object. In this case, this method does not return until the data transfer is complete and is available to be received calling the RemoteGetPicture method.
SyncMode = false. The RemoteTakePicture method will return as soon as a request to the camera to take the picture is sent. The OnRemoteEvent event will be called twice when taking a picture. First with RemoteEventCallbackReleaseStart when the picture is taken and RemoteEventCallbackReleaseComplete when the picture data is available to be read.

If any errors occurs an ECamException exception will be raised.

3.4.23.1 RemoteTakePicture example SyncMode=false

```
procedure TFormRemote.TakeThePicture;
begin
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic);
  //Take a picture. Sync = false
  CamRemote.RemoteTakePicture(false);
end;

//OnRemoteEvent event
procedure TFormRemote.CamRemoteRemoteEvent(Event: RemoteEventCallbackType);
begin
  if (Event = RemoteEventCallbackReleaseComplete) then
  begin
    //First get the thumbnail
    CamRemote.RemoteGetPicture('C:\Temp\Thumb.jpg');
    //Then get the picture
    CamRemote.RemoteGetPicture('C:\Temp\Pic.jpg');
  end;
end;
```

3.4.23.2 RemoteTakePicture example SyncMode=true

```
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic);
  //Take a picture. Sync = true
  num_pics := CamRemote.RemoteTakePicture(true);
  if (num_pics = 2) then
  begin
    //First get the thumbnail
    CamRemote.RemoteGetPicture('C:\Temp\Thumb.jpg');
    //Then get the picture
    CamRemote.RemoteGetPicture('C:\Temp\Pic.jpg');
  end;
```

3.5 TCamRemote events

3.5.1 List of events in TCamRemote

OnEvent
OnGetPictureEvent
OnRemoteEvent
OnRemoteGetPictureEvent
OnRemoteTakePictureEvent
OnViewfinderEvent

3.5.2 TCamRemote.OnEvent

Occurs for camera events (e.g. that the USB connection is lost).

Unit
CamRemote

Syntax:
```
property OnEvent: TNotifyEvent;

TNotifyEvent=procedure(Event:EventCallbackType) of object;
```

Prerequisite:
A successful connection must have been established with the Connect method.

Description
-

3.5.2.1 OnEvent example

```
procedure TFormRemote.CamRemoteEvent(Event: EventCallbackType);
var camera_text   : string;
    text_message  : string;
    message_out   : MessageQueueElementType;
begin¨
  //Log the parameters
  FormRemote.Log('Callback Severity=' + inttostr(Ord(Event.Severity)) +
                 'Event =' + inttostr(Ord(Event.Event)));
  //Do not handle information message, not of interest
  if (Event.Severity = EventSeverityInfo) then
    exit;
  camera_text := 'Camera event-';
  case Event.Severity of
    EventSeverityWarning : text_message := 'Camera warning event';
    EventSeverityClosing : text_message := 'Camera shutdown event';
  end;
  case Event.Event of
    EventBatteryLevelNormal :     camera_text := camera_text + 'Battery level normal';
    EventBatteryLevelWeak :       camera_text := camera_text + 'Battery level weak';
    EventBatteryLevelSafetyLow :  camera_text := camera_text + 'Battery level safety low';
    EventBatteryLevelLB :         camera_text := camera_text + 'Battery level LB';
    EventDialChanged :            camera_text := camera_text + 'Dial changed';
    EventCFGateOpened :           camera_text := camera_text + 'CF gate opened';
    EventBatteryCoverOpened :     camera_text := camera_text + 'Battery cover opened';
    EventConnectionDisappeared :  camera_text := camera_text + 'Camera connection
disappeared';
    EventUnrecoverableError :     camera_text := camera_text + 'Unrecoverable error';
    EventUnkonwnCommandReceived : camera_text := camera_text + 'Unknown command received';
    EventRemoteParamterChanged :  camera_text := camera_text + 'Remote parameter changed';
  end;
  //Log the camera event
  FormRemote.Log(camera_text);
end;
```

3.5.3 TCamRemote.OnRemoteEvent

Occurs for camera remote events (e.g. that remote viewfinder is turned off).

Unit

CamRemote, CamDefines

Syntax:

```
property OnRemoteEvent: TNotifyRemoteEvent;

TNotifyRemoteEvent=procedure(Event:RemoteEventCallbackType) of object;

type
  RemoteEventCallbackType = (RemoteEventCallbackNotUsed,
                             RemoteEventCallbackReleaseStart,
                             RemoteEventCallbackReleaseComplete,
                             RemoteEventCallbackResetHWError,
                             RemoteEventCallbackChangedByUI,
                             RemoteEventCallbackCamReleaseOn,
                             RemoteEventCallbackViewfinderOn,
                             RemoteEventCallbackViewfinderOff);
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.

Description
The interpretation of events are:

RemoteEventCallbackNotUsed	-
RemoteEventCallbackReleaseStart	The request for taking a picture remotely is accepted by the camera.
RemoteEventCallbackReleaseComplete	Picture taken remotely is available to be read using the RemoteGetPicture method.
RemoteEventCallbackResetHWError	A hardware error has occurred.
RemoteEventCallbackChangedByUI	The remote parameters has been changed manually in the camera. Reread new remote parameters using the RemoteGetRemoteParams method.
RemoteEventCallbackCamReleaseOn	The camera shutter release button was pressed. Some camera models do not send this event. **Note:** This event only shows that the camera shutter release button was pressed. It does not show that an image was captured. If the RemoteTakePicture method is executed after this event is received, it will be possible to capture images in the way similar to use the camera shutter release button.
RemoteEventCallbackViewfinderOn	The remote viewfinder is on.
RemoteEventCallbackViewfinderOff	The remote viewfinder is off.

3.5.3.1 OnRemoteEvent example

```
procedure TFormRemote.CamRemoteRemoteEvent(Event: RemoteEventCallbackType);
begin
  FormRemote.Log('Remote callback event=' + inttostr(Ord(Event)));
  //Handle the ReleaseComplete event
  if (Event = RemoteEventCallbackReleaseComplete) then
  begin
    //First get the thumbnail
    CamRemote.RemoteGetPicture('C:\Temp\Thumb.jpg');
    //Then get the picture
    CamRemote.RemoteGetPicture('C:\Temp\Pic.jpg');
  end;
end;
```

3.5.4 TCamRemote.OnGetPictureEvent

Occurs when copying data for a picture stored on the camera to the computer.

Unit
CamRemote

Syntax:
```
property OnGetPictureEvent: TNotifyGetPictureEvent;

TNotifyGetPictureEvent=procedure(PercentageDone:integer) of object;
```

Prerequisite:
A successful connection must have been established with the Connect method.
A successful call to the OpenCameraCollection method, to list pictures on the camera.
The picture is received using the GetPicture method.

Please note the following:
- For each picture transferred from camera to computer this event is called in two cycles (a cycle is a transfer from 0 to 100% of a picture) and not only one.

Description
This event is a callback of progress during file transfer when using the GetPicture method.

3.5.4.1 OnGetPictureEvent example

```
procedure TFormRemote.CamRemoteGetPictureEvent(
  PercentageDone: Integer);
begin
  FormMain.StatusBar.SimpleText := 'Copies picture from camera to computer';
  FormMain.ProgressBar.Position := PercentageDone;
end;
```

3.5.5 TCamRemote.OnRemoteGetPictureEvent

Occurs when copying data for a picture to a destination file.

Unit
CamRemote

Syntax:
```
property OnRemoteGetPictureEvent: TNotifyRemoteGetPictureEvent;

TNotifyRemoteGetPictureEvent=procedure(PercentageDone:integer) of object;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
A picture must have been taken using the RemoteTakePicture method.
The picture is received using the RemoteGetPicture method.

Description
This event is a callback of progress during file transfer when using the RemoteGetPicture method.

3.5.5.1 OnRemoteGetPictureEvent example

```
procedure TFormRemote.CamRemoteRemoteGetPictureEvent(
  PercentageDone: Integer);
begin
  FormRemote.StatusBar.SimpleText := 'Receiving the picture to disc';
  FormRemote.ProgressBar.Position := PercentageDone;
end;
```

3.5.6 TCamRemote.OnRemoteTakePictureEvent

Occurs when transferring data from the camera to the PC.

Unit
CamRemote

Syntax:
```
property OnRemoteTakePictureEvent: TNotifyRemoteTakePictureEvent;

TNotifyRemoteTakePictureEvent=procedure(PercentageDone:integer) of object;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
A picture is taken using the RemoteTakePicture method.

Description
This event is a callback of progress during file transfer from the camera to the PC when using the RemoteTakePicture method (if SyncMode parameter for RemoteTakePicture method is true).

3.5.6.1 OnRemoteTakePictureEvent example

```
procedure TFormRemote.CamRemoteRemoteTakePictureEvent(
  PercentageDone: Integer);
begin
  FormRemote.StatusBar.SimpleText := 'Handles the picture';
  FormRemote.ProgressBar.Position := PercentageDone;
end;
```

3.5.7 TCamRemote.OnViewfinderEvent

Occurs when a remote viewfinder jpeg picture (320x240) is available.

Unit
CamRemote

Syntax:
```
property OnViewfinderEvent: TNotifyViewfinderEvent;

TNotifyViewfinderEvent=procedure(Jpegdata:TMemoryStream) of object;
```

Prerequisite:
A successful connection must have been established with the Connect method.
The camera must be in remote mode, set by the RemoteStart method.
The remote viewfinder must be active, set by the RemoteStartViewfinder.

Description
OnViewfinderEvent is called each time a new remote viewfinder picture is available from the camera.
The camera will take a new remote viewfinder picture, when this event exits.

3.5.7.1 OnRemoteViewfinder example

```
procedure TFormRemote.CamRemoteViewfinderEvent(Jpegdata: TMemoryStream);
var jpg : TJpegImage;
begin
  jpg := TJpegImage.Create;
  //Get the 320x240 picture and update the image on the dialogue
  try
    jpg.LoadFromStream(Jpegdata);
    FormViewFinder.Image.Picture.Assign(jpg);
  finally
    jpg.Free;
  end;
end;
```

3.6 TCamRemote types

3.6.1 ConnectInfoType

Unit
CamDefines

Syntax:
```
  CamModType = (CamModNone,
                CamModPowerShot,
                CamModEOS);

  ConnectInfoType = record
                      CameraModel     : CamModType;
                      CameraModelName : string;
                      OwnerName       : string;
                    end;
```

Description
Used be the Connect method to return information about the connected camera.

3.6.2 EventCallbackType

Unit
CamDefines

Syntax:
```
  EventCallbackType = record
                        Severity : EventSeverityType;
                        Event    : EventEnumType;
                      end;

  EventSeverityType = (EventSeverityNotUsed,
                       EventSeverityInfo,
                       EventSeverityWarning,
                       EventSeverityClosing);

  EventEnumType     = (EventNotUsed,
                       EventBatteryLevelNormal,
                       EventBatteryLevelWeak,
                       EventBatteryLevelSafetyLow,
                       EventBatteryLevelLB,
                       EventDialChanged,
                       EventCFGateOpened,
                       EventBatteryCoverOpened,
                       EventConnectionDisappeared,
                       EventUnrecoverableError,
                       EventUnkonwnCommandReceived,
                       EventRemoteParamterChanged);
```

Description
EventCallbackType is used as a parameter to the OnEvent event. The interpretation of EventSeverityType is:

`EventSeverityInfo`	The event is of informational type, and is not critical.
`EventSeverityWarning`	The event is more severe and can cause problems if no measures are taken. Example of a severe event is low/weak battery level.
`EventSeverityClosing`	The event is critical and the connection to the camera is immediately closed.

3.6.3 ImageListType

Unit
CamDefines

Syntax:
```
  ImageListType = array of ImageDataType;

  ImageDataType = record
```

```
                    Name      : string;
                    Thumbnail : TJPEGImage;
{$IFDEF DEXIF}
                    EXIF      : TImgData;
{$ENDIF}
                  end;
```

Description
Used be the OpenCameraCollection method to return information about pictures stored on the camera.
ImageListType returns a list of objects, where each objects includes meta data for each picture on the camera. The meta data (ImageDataType) includes the name, a thumbnail and optionally EXIF information. EXIF information is only available for RAW pictures and requires that the dEXIF component is installed.

3.6.4 RemoteEventCallbackType

Unit
CamDefines

Syntax:
```
RemoteEventCallbackType = (RemoteEventCallbackNotUsed,
                           RemoteEventCallbackReleaseStart,
                           RemoteEventCallbackReleaseComplete,
                           RemoteEventCallbackResetHWError,
                           RemoteEventCallbackChangedByUI,
                           RemoteEventCallbackCamReleaseOn,
                           RemoteEventCallbackViewfinderOn,
                           RemoteEventCallbackViewfinderOff);
```

Description
RemoteEventCallbackType is used as a parameter to the OnRemoteEvent event. The interpretation of RemoteEventCallbackType is:

`RemoteEventCallbackReleaseStart`	A take picture remote request is sent to the camera. Only valid if SyncMode is false when using the RemoteTakePicture method.
`RemoteEventCallbackReleaseComplete`	Pictures that have been remotely taken can be received using the RemoteGetPicture method. Only valid if SyncMode is false when using the RemoteTakePicture method.
`RemoteEventCallbackResetHWError`	Hardware error.
`RemoteEventCallbackChangedByUI`	The user has change remote parameters manually in the camera. Use the RemoteGetRemoteParams method to update current used remote parameters.
`RemoteEventCallbackCamReleaseOn`	The user has manually taken a picture by pressing the take picture button on the camera. Receive the taken picture using the RemoteGetPicture method.
`RemoteEventCallbackViewfinderOn`	The remote viewfinder is on.
`RemoteEventCallbackViewfinderOff`	The remote viewfinder is off.

3.6.5 RemoteFuncType

Unit
CamDefines

Syntax:
```
RemoteFuncType = record
                   DoSupportZoom               : boolean;
                   DoSupportShootingPara       : boolean;
                   DoSupportViewfinder         : boolean;
                   ReqViewfinderOffWhenShooting : boolean;
```

```
    DoSupportAfLockUnlock        : boolean;
    RemoteParamSupported         : RemoteReleaseAvailParametersType;
  end;
```

Description
Used be the RemoteStart method to return information about the connected camera. The record is interpreted as follows:

`DoSupportZoom`	Is true if remote zooming is supported.
`DoSupportShootingPara`	Is true if remote parameters are supported.
`DoSupportViewfinder`	Is true if remote viewfinder is supported.
`ReqViewfinderOffWhenShooting`	Is true if remote viewfinder must be off when taking a picture remotely. Is false if viewfinder can be on when taking picture remotely using the RemoteTakePicture method.
`DoSupportAfLockUnlock`	Is true if AF lock/unlock is supported.
`RemoteParamSupported`	Supported remote parameters probed during execution of the RemoteStart method. Data is valid if DoSupportShootingPara is true.

3.6.6 RemoteReleaseAvailParametersType

Unit
CamDefines

Syntax:
```
type RemoteReleaseAvailParametersType =
  record
    CompQuality          : array[RemoteFormatQualityType] of boolean;
    ImageSize            : array[RemoteFormatSizeType] of boolean;
    StrobeSetting        : array[RemoteFormatFlashType] of boolean;
    StrobeCompSetting    : array[RemoteFormatFlashCompType] of boolean;
    ImageMode            : array[RemoteFormatShootingModeType] of boolean;
    MLWeiMode            : array[RemoteFormatMLWeiType] of boolean;
    AFDistance           : array[RemoteFormatAFDistType] of boolean;
    WhiteBalanceSetting  : array[RemoteFormatWBType] of boolean;
    Contrast             : array[RemoteFormatLevelType] of boolean;
    ColorGain            : array[RemoteFormatLevelType] of boolean;
    Sharpness            : array[RemoteFormatLevelType] of boolean;
    ISO                  : array[RemoteFormatISOType] of boolean;
    Av                   : array[RemoteFormatAVType] of boolean;
    Tv                   : array[RemoteFormatTVType] of boolean;
    ExposureCompensation : array[RemoteFormatExposureCompType] of boolean;
    PhotoEffect          : array[RemoteFormatPhotoEffectType] of boolean;
    Beep                 : array[RemoteFormatBeepType] of boolean;
   end;
```

Description
Remote parameter capability type. True indicates that the parameter is supported. False that it is not supported.

3.6.7 RemoteReleaseParametersType

Unit
CamDefines

Syntax:
```
type RemoteReleaseParametersType =
  record
    CompQuality          : RemoteFormatQualityType;
    ImageSize            : RemoteFormatSizeType;
    StrobeSetting        : RemoteFormatFlashType;
    StrobeCompSetting    : RemoteFormatFlashCompType;
    ImageMode            : RemoteFormatShootingModeType;
    MLWeiMode            : RemoteFormatMLWeiType;
    AFDistance           : RemoteFormatAFDistType;
    WhiteBalanceSetting  : RemoteFormatWBType;
    Contrast             : RemoteFormatLevelType;
    ColorGain            : RemoteFormatLevelType;
```

```
    Sharpness           : RemoteFormatLevelType;
    ISO                 : RemoteFormatISOType;
    Av                  : RemoteFormatAVType;
    Tv                  : RemoteFormatTVType;
    ExposureCompensation : RemoteFormatExposureCompType;
    PhotoEffect         : RemoteFormatPhotoEffectType;
    Beep                : RemoteFormatBeepType;
  end;

type RemoteFormatQualityType = (RemoteFormatQualityNotUsed,
                                RemoteFormatQualityNormal,
                                RemoteFormatQualityFine,
                                RemoteFormatQualityLossless,
                                RemoteFormatQualitySuperfine,
                                RemoteFormatQualityRAW);

type RemoteFormatSizeType = (RemoteFormatSizeNotUsed,
                             RemoteFormatSizeLarge,
                             RemoteFormatSizeMedium,
                             RemoteFormatSizeSmall,
                             RemoteFormatSizeMedium2,
                             RemoteFormatSizeMedium1);

type RemoteFormatShootingModeType = (RemoteFormatShootingModeNotUsed,
                                     RemoteFormatShootingModeAuto,
                                     RemoteFormatShootingModeManual,
                                     RemoteFormatShootingModeFarScene,
                                     RemoteFormatShootingModeFastShutter,
                                     RemoteFormatShootingModeSlowShutter,
                                     RemoteFormatShootingModeNightScene,
                                     RemoteFormatShootingModeGrayScene,
                                     RemoteFormatShootingModeSepia,
                                     RemoteFormatShootingModePortrait,
                                     RemoteFormatShootingModeSport,
                                     RemoteFormatShootingModeMacro,
                                     RemoteFormatShootingModeBW,
                                     RemoteFormatShootingModePanFocus,
                                     RemoteFormatShootingModeVivid,
                                     RemoteFormatShootingModeNeutral,
                                     RemoteFormatShootingModeProgram,
                                     RemoteFormatShootingModeTV,
                                     RemoteFormatShootingModeAV);

type RemoteFormatExposureCompType = (RemoteFormatExposureCompNotUsed,
                                     RemoteFormatExposureComp200Plus,
                                     RemoteFormatExposureComp166Plus,
                                     RemoteFormatExposureComp133Plus,
                                     RemoteFormatExposureComp100Plus,
                                     RemoteFormatExposureComp066Plus,
                                     RemoteFormatExposureComp033Plus,
                                     RemoteFormatExposureComp0,
                                     RemoteFormatExposureComp033Minus,
                                     RemoteFormatExposureComp066Minus,
                                     RemoteFormatExposureComp100Minus,
                                     RemoteFormatExposureComp133Minus,
                                     RemoteFormatExposureComp166Minus,
                                     RemoteFormatExposureComp200Minus);

type RemoteFormatWBType = (RemoteFormatWBNotUsed,
                           RemoteFormatWBAuto,
                           RemoteFormatWBDaylight,
                           RemoteFormatWBCloudy,
                           RemoteFormatWBTungsten,
                           RemoteFormatWBFluorscent,
                           RemoteFormatWBFlash,
                           RemoteFormatWBPreset,
                           RemoteFormatWBFluorescentLight,
                           RemoteFormatWBCustom);

type RemoteFormatAFDistType = (RemoteFormatAFDistNotUsed,
                               RemoteFormatAFDistManual,
                               RemoteFormatAFDistAuto,
                               RemoteFormatAFDistUnknown,
                               RemoteFormatAFDistZFCloseUp,
                               RemoteFormatAFDistZFShortestDistance,
                               RemoteFormatAFDistZFShortDistance,
                               RemoteFormatAFDistZFMediumDistance,
                               RemoteFormatAFDistZFFarDistance);

type RemoteFormatFlashType = (RemoteFormatFlashNotUsed,
                              RemoteFormatFlashOff,
```

```
                                RemoteFormatFlashAuto,
                                RemoteFormatFlashOn,
                                RemoteFormatFlashRedEye,
                                RemoteFormatFlashSlowSync,
                                RemoteFormatFlashAutoRedEye,
                                RemoteFormatFlashOnRedEye);

      type RemoteFormatFlashCompType = (RemoteFormatFlashCompNotUsed,
                                   RemoteFormatFlashComp200Plus,
                                   RemoteFormatFlashComp166Plus,
                                   RemoteFormatFlashComp133Plus,
                                   RemoteFormatFlashComp100Plus,
                                   RemoteFormatFlashComp066Plus,
                                   RemoteFormatFlashComp033Plus,
                                   RemoteFormatFlashComp0,
                                   RemoteFormatFlashComp033Minus,
                                   RemoteFormatFlashComp066Minus,
                                   RemoteFormatFlashComp100Minus,
                                   RemoteFormatFlashComp133Minus,
                                   RemoteFormatFlashComp166Minus,
                                   RemoteFormatFlashComp200Minus);

      type RemoteFormatMLWeiType = (RemoteFormatMLWeiNotUsed,
                                RemoteFormatMLWeiCenterWeighted,
                                RemoteFormatMLWeiSpot,
                                RemoteFormatMLWeiAveraging,
                                RemoteFormatMLWeiEvaluative,
                                RemoteFormatMLWeiPartial,
                                RemoteFormatMLWeiCenterwWeightedAveraging);

      //Remote Contrast, Color Gain, Sharpness
      type RemoteFormatLevelType = (RemoteFormatLevelNotUsed,
                                RemoteFormatLevelLow,
                                RemoteFormatLevelDefault,
                                RemoteFormatLevelHigh);

      type RemoteFormatISOType = (RemoteFormatISONotUsed,
                               RemoteFormatISOAuto,
                               RemoteFormatISO50,
                               RemoteFormatISO100,
                               RemoteFormatISO200,
                               RemoteFormatISO400,
                               RemoteFormatISO800,
                               RemoteFormatISO1600,
                               RemoteFormatISO3200);

      type RemoteFormatTVType = (RemoteFormatTVNotUsed,
                              RemoteFormatTV30sec,
                              RemoteFormatTV15sec,
                              RemoteFormatTV8sec,
                              RemoteFormatTV4sec,
                              RemoteFormatTV2sec,
                              RemoteFormatTV1sec,
                              RemoteFormatTV0sec5,
                              RemoteFormatTV1_4,
                              RemoteFormatTV1_8,
                              RemoteFormatTV1_15,
                              RemoteFormatTV1_30,
                              RemoteFormatTV1_60,
                              RemoteFormatTV1_125,
                              RemoteFormatTV1_250,
                              RemoteFormatTV1_500,
                              RemoteFormatTV1_1000,
                              RemoteFormatTV1_2000,
                              RemoteFormatTV1_4000,
                              RemoteFormatTV1_8000,
                              RemoteFormatTV1_16000);

      type RemoteFormatAVType = (RemoteFormatAVNotUsed,
                              RemoteFormatAV1_0,
                              RemoteFormatAV1_4,
                              RemoteFormatAV2_0,
                              RemoteFormatAV2_8,
                              RemoteFormatAV4_0,
                              RemoteFormatAV5_6,
                              RemoteFormatAV8_0,
                              RemoteFormatAV11_0,
                              RemoteFormatAV16_0,
                              RemoteFormatAV22_0,
                              RemoteFormatAV32_0,
                              RemoteFormatAV45_0,
```

```
                                RemoteFormatAV64_0,
                                RemoteFormatAV91_0);

  type RemoteFormatPhotoEffectType = (RemoteFormatPhotoEffectNotUsed,
                                      RemoteFormatPhotoEffectOff,
                                      RemoteFormatPhotoEffectVivid,
                                      RemoteFormatPhotoEffectNeutral,
                                      RemoteFormatPhotoEffectLowSharpening,
                                      RemoteFormatPhotoEffectSepia,
                                      RemoteFormatPhotoEffectBW);

  type RemoteFormatBeepType = (RemoteFormatBeepNotUsed,
                               RemoteFormatBeepOn,
                               RemoteFormatBeepOff);
```

Description
Type used to get and set remote parameters to/from the camera. Is used by the RemoteSetRemoteParams and RemoteGetRemoteParams methods.

3.6.8 RemoteZoomCapabilityType

Unit
CamDefines

Syntax:

```
  RemoteZoomCapabilityType = record
                               CurrentZoomPos    : integer;
                               MaxOpticalZoomPos : integer;
                               MaxZoomPos        : integer;
                             end;
```

Description
RemoteZoomCapabilityType is used by the RemoteGetZoomPos method. The interpretation of RemoteZoomCapabilityType is:

`CurrentZoomPos`	The current zoom position on the camera.
`MaxOpticalZoomPos`	The maximum optical zoom position.
`MaxZoomPos`	The maximum digital zoom position.

0 < MaxOpticalZoomPos < MaxZoomPos

3.7 TCamRemote error codes

```
{+//--------------------------------------------------------------------}
{-Error Code Masks }
{=--------------------------------------------------------------------}
const
  cdERROR_ISSPECIFIC_MASK = $80000000;
const
  cdERROR_COMPONENTID_MASK = $7F000000;
const
  cdERROR_RESERVED_MASK = $00FF0000;
const
  cdERROR_ERRORID_MASK = $0000FFFF;

{+//--------------------------------------------------------------------}
{-Base Component IDs }
{=--------------------------------------------------------------------}
const
  cdERROR_CLIENT_COMPONENTID = $01000000;
const
  cdERROR_LLSDK_COMPONENTID = $02000000;
const
  cdERROR_HLSDK_COMPONENTID = $03000000;
const
  cdERROR_PROPERTY_PARSING_COMPONENTID = $04000000;
const
  cdERROR_VIEW_DEVELOPMENT_COMPONENTID = $05000000;
const
  cdERROR_VIEW_DECODING_COMPONENTID = $06000000;
const
  cdERROR_COLOR_MAPPING_COMPONENTID = $07000000;
```

```
const
  cdERROR_PICTURE_COLLECTION_COMPONENTID = $08000000;
const
  cdERROR_SETUP_COMPONENTID = $09000000;
const
  cdERROR_IWRAP_COMPONENTID = $0A000000;
const
  cdERROR_PSUSD_COMPONENTID = $0B000000;
const
  cdERROR_CDSDK_COMPONENTID = $0C000000;
const
  cdERROR_RDSDK_COMPONENTID = $0D000000;


{+//--------------------------------------------------------------------}
{-Functin Success Code }
{=--------------------------------------------------------------------}
const
  cdOK = $00000000;

{+//--------------------------------------------------------------------}
{-Generic Error IDs }
{=--------------------------------------------------------------------}
{+// Miscellaneous errors*/ }
const
  cdUNIMPLEMENTED = $00000001;
const
  cdINTERNAL_ERROR = $00000002;
const
  cdMEM_ALLOC_FAILED = $00000003;
const
  cdMEM_FREE_FAILED = $00000004;
const
  cdOPERATION_CANCELLED = $00000005;
const
  cdINCOMPATIBLE_VERSION = $00000006;
const
  cdNOT_SUPPORTED = $00000007;
const
  cdUNEXPECTED_EXCEPTION = $00000008;
const
  cdPROTECTION_VIOLATION = $00000009;
const
  cdMISSING_SUBCOMPONENT = $0000000A;
const
  cdSELECTION_UNAVAILABLE = $0000000B;

{+// File errors*/ }
const
  cdFILE_IO_ERROR = $00000020;
const
  cdFILE_TOO_MANY_OPEN = $00000021;
const
  cdFILE_NOT_FOUND = $00000022;
const
  cdFILE_OPEN_ERROR = $00000023;
const
  cdFILE_CLOSE_ERROR = $00000024;
const
  cdFILE_SEEK_ERROR = $00000025;
const
  cdFILE_TELL_ERROR = $00000026;
const
  cdFILE_READ_ERROR = $00000027;
const
  cdFILE_WRITE_ERROR = $00000028;
const
  cdFILE_PERMISSION_ERROR = $00000029;
const
  cdFILE_DISK_FULL_ERROR = $0000002A;
const
  cdFILE_ALREADY_EXISTS = $0000002B;
const
  cdFILE_FORMAT_UNRECOGNIZED = $0000002C;
const
  cdFILE_DATA_CORRUPT = $0000002D;
const
  cdFILE_NAMING_NA = $0000002E;
```

```
{+// Directory errors*/ }
const
  cdDIR_NOT_FOUND = $00000040;
const
  cdDIR_IO_ERROR = $00000041;
const
  cdDIR_ENTRY_NOT_FOUND = $00000042;
const
  cdDIR_ENTRY_EXISTS = $00000043;
const
  cdDIR_NOT_EMPTY = $00000044;

{+// Property errors*/ }
const
  cdPROPERTIES_UNAVAILABLE = $00000050;
const
  cdPROPERTIES_MISMATCH = $00000051;
const
  cdPROPERTIES_NOT_LOADED = $00000053;

{+// Function Parameter errors*/ }
const
  cdINVALID_PARAMETER = $00000060;
const
  cdINVALID_HANDLE = $00000061;
const
  cdINVALID_POINTER = $00000062;
const
  cdINVALID_INDEX = $00000063;
const
  cdINVALID_LENGTH = $00000064;
const
  cdINVALID_FN_POINTER = $00000065;
const
  cdINVALID_SORT_FN = $00000066;

{+// Device errors*/ }
const
  cdDEVICE_NOT_FOUND = $00000080;
const
  cdDEVICE_BUSY = $00000081;
const
  cdDEVICE_INVALID = $00000082;
const
  cdDEVICE_EMERGENCY = $00000083;
const
  cdDEVICE_MEMORY_FULL = $00000084;
const
  cdDEVICE_INTERNAL_ERROR = $00000085;
const
  cdDEVICE_INVALID_PARAMETER = $00000086;
const
  cdDEVICE_NO_DISK = $00000087;
const
  cdDEVICE_DISK_ERROR = $00000088;
const
  cdDEVICE_CF_GATE_CHANGED = $00000089;
const
  cdDEVICE_DIAL_CHANGED = $0000008A;
const
  cdDEVICE_NOT_INSTALLED = $0000008B;
const
  cdDEVICE_STAY_AWAKE = $0000008C;
const
  cdDEVICE_NOT_RELEASED = $0000008D;

{+// Stream errors*/ }
const
  cdSTREAM_IO_ERROR = $000000A0;
const
  cdSTREAM_NOT_OPEN = $000000A1;
const
  cdSTREAM_ALREADY_OPEN = $000000A2;
const
  cdSTREAM_OPEN_ERROR = $000000A3;
const
  cdSTREAM_CLOSE_ERROR = $000000A4;
```

```
const
  cdSTREAM_SEEK_ERROR = $000000A5;
const
  cdSTREAM_TELL_ERROR = $000000A6;
const
  cdSTREAM_READ_ERROR = $000000A7;
const
  cdSTREAM_WRITE_ERROR = $000000A8;
const
  cdSTREAM_PERMISSION_ERROR = $000000A9;
const
  cdSTREAM_COULDNT_BEGIN_THREAD = $000000AA;
const
  cdSTREAM_BAD_OPTIONS = $000000AB;
const
  cdSTREAM_END_OF_STREAM = $000000AC;

{+// Communications errors*/ }
const
  cdCOMM_PORT_IS_IN_USE = $000000C0;
const
  cdCOMM_DISCONNECTED = $000000C1;
const
  cdCOMM_DEVICE_INCOMPATIBLE = $000000C2;
const
  cdCOMM_BUFFER_FULL = $000000C3;
const
  cdCOMM_USB_BUS_ERR = $000000C4;

{+// Lock/Unlock*/ }
const
  cdUSB_DEVICE_LOCK_ERROR = $000000D0;
const
  cdUSB_DEVICE_UNLOCK_ERROR = $000000D1;

{+// STI/WIA*/ }
const
  cdSTI_UNKNOWN_ERROR = $000000E0;
const
  cdSTI_INTERNAL_ERROR = $000000E1;
const
  cdSTI_DEVICE_CREATE_ERROR = $000000E2;
const
  cdSTI_DEVICE_RELEASE_ERROR = $000000E3;
const
  cdDEVICE_NOT_LAUNCHED = $000000E4;

const
  cdENUM_NA = $000000F0;
const
  cdINVALID_FN_CALL = $000000F1;
const
  cdHANDLE_NOT_FOUND = $000000F2;
const
  cdINVALID_ID = $000000F3;
const
  cdWAIT_TIMEOUT_ERROR = $000000F4;

{+// PTP*/ }
const
  cdSESSION_NOT_OPEN = $00002003;
const
  cdINVALID_TRANSACTIONID = $00002004;
const
  cdINCOMPLETE_TRANSFER = $00002007;
const
  cdINVALID_STRAGEID = $00002008;
const
  cdDEVICEPROP_NOT_SUPPORTED = $0000200A;
const
  cdINVALID_OBJECTFORMATCODE = $0000200B;
const
  cdSELF_TEST_FAILED = $00002011;
const
  cdPARTIAL_DELETION = $00002012;
const
  cdSPECIFICATION_BY_FORMAT_UNSUPPORTED = $00002014;
```

```
const
  cdNO_VALID_OBJECTINFO = $00002015;
const
  cdINVALID_CODE_FORMAT = $00002016;
const
  cdUNKNOWN_VENDER_CODE = $00002017;
const
  cdCAPTURE_ALREADY_TERMINATED = $00002018;
const
  cdINVALID_PARENTOBJECT = $0000201A;
const
  cdINVALID_DEVICEPROP_FORMAT = $0000201B;
const
  cdINVALID_DEVICEPROP_VALUE = $0000201C;
const
  cdSESSION_ALREADY_OPEN = $0000201E;
const
  cdTRANSACTION_CANCELLED = $0000201;
const
  cdSPECIFICATION_OF_DESTINATION_UNSUPPORTED = $00002020;
const
  cdUNKNOWN_COMMAND = $0000A001;
const
  cdOPERATION_REFUSED = $0000A005;
const
  cdLENS_COVER_CLOSE = $0000A006;

const
  cdLAST_GENERIC_ERROR_PLUS_ONE = $000000F5;
```

Index

- A -

- B -

- C -

- D -

- E -

- G -

- I -

- L -

- M -

- N -

- O -

- P -

- R -

- S -

- T -

- W -

Definition of Photography on the net

The art or process of producing images by the action of light on surfaces sensitized by chemical processes.

A process by which chemically sensitized surfaces are exposed to light (photo) and retain an image (graph) of what is exposed. Methods may be very simple to highly complex. Camera are usually used with adjustable lenses (apertures) and controlled light levels on light sensitive film. The film is then processed (developed) and the image is "fixed" (made permanent). The image (a negative) is transferred onto treated papers, enlarged and processed with chemicals in a "dark room" to make the photographs (also called prints).

Developed in the second half of the 19th century, this development was very important in astronomy. The first pictures of space were taken around 1840, but the methods of photography weren't important in astronomy until about twenty years later. But when they were used, they told us things we couldn't see before. Photographs of the Moon were used to draw atlases, sunspots were more easily recorded, details of nebulae and stars were found. In 1882, Sir David Gill photographed a comet and discovered that the picture showed tons of stars . . . a great way to map the sky.

A term which comes from the Greek words photos (light) and graphos (drawing). A photograph is made with a camera by exposing film to light in order to create a negative. The negative is then used in the darkroom to print a photograph (positive) onto light-sensitive paper.