TCamRemote
ActiveX component
version 4.5

TCamRemote

The ActiveX component to handle PowerShot/EOS digital cameras

---

by Hans-David Alkenius

You have Canon digital camera(s)? Want to use the camera(s) as picture source to your computer? Maybe want to create a movie or handle RAW pictures?

The TCamRemote component for ActiveX enables you to handle PowerShot and/or EOS digital camera(s) within your development platform.

TCamRemote

Printed: november 2006 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

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 connected camera(s),
- handle multiple cameras. It is possible to handle one camera at a time or many camera at once if the cameras are of different models (e.g. a PowerShot S40 a PowerShot S70 and a EOS 10D).
- develop RAW-pictures with development parameters (e.g. change whitebalance) to 8 or 16 bits Tiffs.

1.2 Description

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

Description

The TCamRemote ActiveX version is based on the TCamRemote Delphi component which used to handle one or several Canon PowerShot and/or EOS digital cameras.

Updates

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

Development Requirements
Any development platform that supports ActiveX such as Visual Basic, Visual C++, VB.NET, C#, ASP, ASP.NET, Access, Borland C++ Builder, PowerBuilder, FoxPro.

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. EOS-DLL requires Windows 2000 or 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.
- Upload pictures to the camera.
- Use stream to avoid temporary files (for e.g. get thumbnails).

RAW-development:

- Add many more development parameters.
- Add PowerShot development functions.
- Set Whitebalance by clicking a preview.
- Generate preview to file or TImage/TPicture/TBitmap. Different sizes and qualities versus speed.
- Save to other format than TIFF.
- Store EXIF in JPEG/TIFF.
- Store and get ICC profiles for both 8 and 16 bits.
- Automatically rotate developed pictures correctly, using orientation information in the EXIF-data.

How it happened...

From the beginning (year 2000) I developed a tool called Cam4you utilities. Cam4you utilities homepage is http://alkenius.no-ip.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. At the end of 2006 I decided to create a TCamRemote ActiveX version based on the Delphi VCL version. The main approach was to create a TCamRemote ActiveX and make sure that all the methods, properties and events are included from the VCL TCamRemote interfaces.

1.3 License

As a proprietary product, TCamRemote is protected by copyright laws. At all times Alkenius Systems 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 Alkenius Systems 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 6.

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 Alkenius Systems.

AGREEMENT PERTAINING TO THE RELEASE OF SOURCE CODE by Alkenius Systems 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 Alkenius Systems has provided.

CHANGES TO SOURCE CODE

Alkenius Systems 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. Alkenius Systems 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 Alkenius Systems liable, directly or indirectly, for any changes made to the source, including changes which Recipient has made based on advice or suggestions provided by Alkenius Systems. The Recipent shall get back the payment for the Software, if critical errors found in the Software not can be solved by Alkenius Systems. Recipent will not hold Alkenius Systems liable for any hardware problems when using the Software.

1.4 Evaluation of TCamRemote

TCamRemote is downloaded as a trial period version. The trial period is 60 days from the time the TCamRemote component is used for the first time. A program which uses TCamRemote trial version will also have a trial period of 60 days.

It is possible to extend the evaluation period. Send a request to the owner of TCamRemote.

The TCamRemote trial version can be registered by entering a registration code and name in the RegCodeOCX and RegNameOCX design component fields. When TCamRemote is registered all applications using the registered version will have no limitation and will have no time trial restrictions.

Part II

2 Overview

2.1 Overview ActiveX

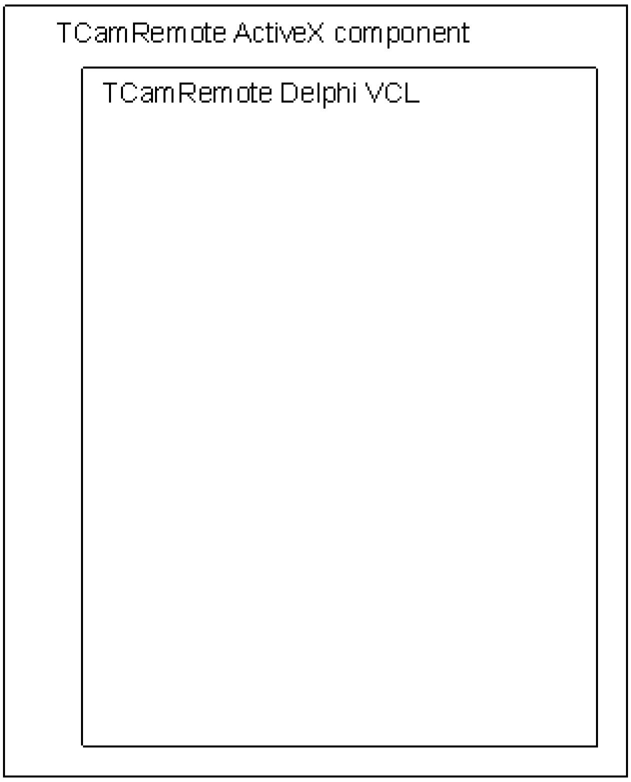
The TCamRemote ActiveX version is a wrapper for the TCamRemote Delphi VCL version. In Delphi it is possible to create ActiveX version of VCL components, but only the interfaces that fulfills the requirements on data types are included in the ActiveX component. Only simple data types (e.g. integer, string etc) are supported by ActiveX. Since the TCamRemote VCL version, uses more complex data types, specially data records, is the TCamRemote VCL interfaces wrapped in another VCL component only used for creating the ActiveX version, see picture below.

The names of methods and data types are shared as much as possible between the Delphi VCL version and the ActiveX version, but the ActiveX version have a few addons. A function in the Delphi VCL version returning complex data records are first changed to a procedure. Then several interface methods with a prefix name same as the original function, and appended with a '_' and then the data. In some cases 'GetXXX' functions are created. Below a few examples:
The Connect 4 functions returns complex data structures in the Delphi VCL version. In the ActiveX version these data structures can be received using the Connect_CameraModel, 6 Connect_OwnerName 7 etc.

Therefore may the TCamRemote VCL version manual be used as a compliment, since the information and data is shared as much as possible.

The TCamRemote ActiveX version is based on a TCamRemote VCL version which includes the PS-DLL interface for PowerShot support and EOS-DLL interface for EOS support. The EOS-OLD-DLL interface is not supported by the ActiveX version.

Depending on which cameras TCamRemote supports, required run-time DLLs 2 must be reached by the application using TCamRemote.



2.2 Supported cameras

The following cameras are supported:

PowerShot:

Camera model	Remote handling supported	Remote viewfinder supported	Picture download
PowerShot A10	yes	no	yes
PowerShot A20	yes	no	yes
PowerShot A30	yes	no	yes
PowerShot A40	yes	no	yes
PowerShot A60	yes	yes	yes
PowerShot A70	yes	yes	yes
PowerShot A75	yes	yes	yes
PowerShot A80	yes	yes	yes
PowerShot A85	yes	yes	yes
PowerShot A95	yes	yes	yes
PowerShot A100	yes	yes	yes
PowerShot A200	yes	yes	yes
PowerShot A300	yes	yes	yes
PowerShot A310	yes	yes	yes
PowerShot A400	yes	yes	yes
PowerShot A510	yes	yes	yes
PowerShot A520	yes	yes	yes
PowerShot A620	yes	yes	no
PowerShot S1 IS	yes	yes	yes
PowerShot S2 IS	yes	yes	yes
PowerShot S3 IS	yes	yes	no
PowerShot S10	no	no	yes
PowerShot S20	no	no	yes
PowerShot S30	yes	yes	yes
PowerShot S40	yes	yes	yes
PowerShot S45	yes	yes	yes
PowerShot S50	yes	yes	yes
PowerShot S60	yes	yes	yes
PowerShot S70	yes	yes	yes
PowerShot S80	yes	yes	no
PowerShot S100, IXY DIGITAL, DIGITAL IXUS	yes	no	yes
PowerShot S110, IXY DIGITAL 200, DIGITAL IXUS v	yes	no	yes
PowerShot S200, IXY DIGITAL 200a, DIGITAL IXUS v2	yes	yes	yes
PowerShot S230, IXY DIGITAL 320, DIGITAL IXUS v3	yes	yes	yes
PowerShot S300, IXY DIGITAL 300, DIGITAL IXUS 300	yes	no	yes
PowerShot S330, IXY DIGITAL 300a, DIGITAL IXUS 330	no	no	no

Notes regarding PowerShot support:

- PowerShot A410, A420, A430, A530, A540, A610, A630, A700, A710 IS, SD30, SD40, SD430, SD450, SD550, SD600, SD630, SD700 IS, SD800 IS, and SD900 are **not** supported by TCamRemote.
- Support for PowerShot A640 and G7 will hopefully be added December 2006.

EOS:
EOS 1D Mark II, EOS 20D, EOS 1Ds Mark II
EOS Kiss Digital N/350D/REBEL XT
EOS 5D, EOS 1D Mark II N, EOS 30D
EOS Kiss Digital X/400D/REBEL XTi

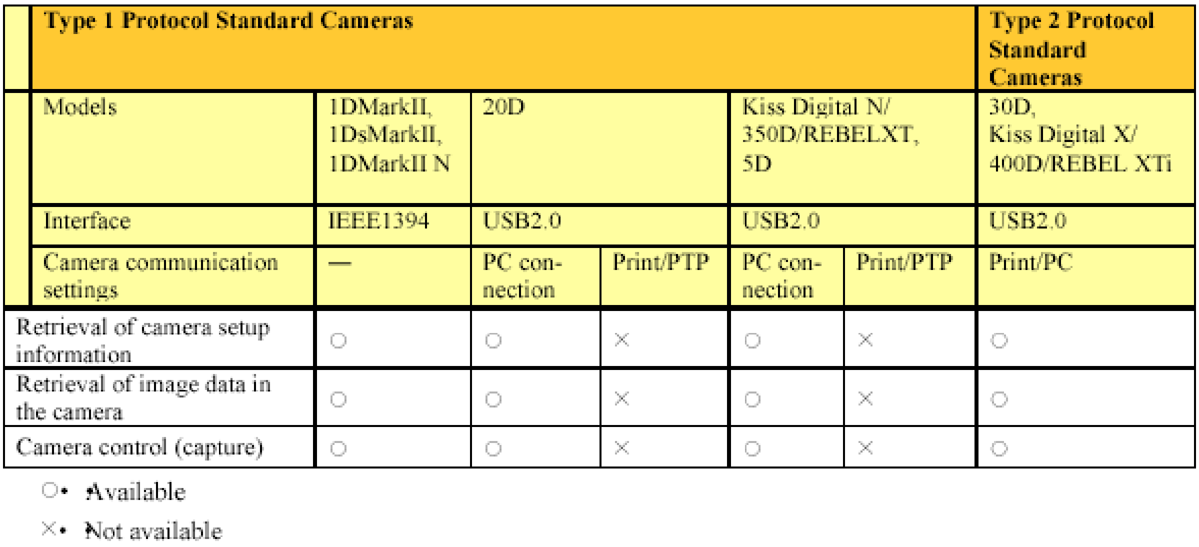
2.3 Camera protocol

Protocol for Remote Connection
Two types of protocol are used by EOS Digital to connect to a host PC. TCamRemote applications can basically communicate with remotely connected cameras without any awareness of the difference between protocols.

Type 1 (Legacy Protocol)
Legacy protocol is an original protocol from Canon for connections between a host PC and camera. This protocol is incorporated into cameras up to EOS5D and in EOS (EOS1 series) cameras with an IEEE1394 interface. A special device driver for the connected camera must be installed on the host PC in order to connect using this protocol. Be sure to install this driver beforehand from the CD-ROM supplied with Canon cameras or by downloading from Canon's homepage. Cameras which use a Type 1 protocol as standard such as EOS 1DmarkII N are called "Type 1 protocol standard cameras" in this manual.

Type 2 (PTP)
PTP is an abbreviation of "Picture Transfer Protocol." PTP is a standard protocol used to transfer images to a PC. This protocol is incorporated in EOS digital cameras that include a USB interface starting with EOS Kiss Digital N (EOS 350D/REBEL XT). A device driver for each model is unnecessary when connecting to an OS that supports PTP. (However, a device driver for making PTP connections is required when using an OS which does not support PTP as standard such as Windows 2000. This driver can be obtained from the CD-ROM supplied with Canon cameras or by downloading from Canon's homepage.) Type 1 protocol has been eliminated from cameras with a USB interface starting from EOS30D and Type 2 protocol is utilized as that standard. Cameras that use Type 2 protocol as standard such as EOS30D are called "Type 2 protocol standard cameras" in this manual. EOS Kiss Digital N , 350D, REBELXT, and EOS 5D model cameras come shipped from the factory with communications set for [Print/PTP] but functions that support PC connections are limited. For example, capture-related features cannot be used. Since these cameras use [PC connection] (Type 1 protocol) as the standard for connecting to a PC, they are Type 1 protocol standard cameras.

Support by model
The following table shows the protocol which can be used for each model when controlling a remotely connected camera. Be sure to set the communication settings of the camera as follows.



	Type 1 Protocol Standard Cameras					Type 2 Protocol Standard Cameras
Models	1DMarkII, 1DsMarkII, 1DMarkII N	20D		Kiss Digital N/ 350D/REBELXT, 5D		30D, Kiss Digital X/ 400D/REBEL XTi
Interface	IEEE1394	USB2.0		USB2.0		USB2.0
Camera communication settings	—	PC connection	Print/PTP	PC connection	Print/PTP	Print/PC
Retrieval of camera setup information	○	○	×	○	×	○
Retrieval of image data in the camera	○	○	×	○	×	○
Camera control (capture)	○	○	×	○	×	○

○• Available

×• Not available

2.4 Revision History

Version 4.5

- The TCamRemote ActiveX version created.
- The GetFirmwareVersion method removed, since it caused several unknown raised exceptions.
- The sharpness remote parameter is correctly set by TCamRemote. The high and low values were previuosly mixed by mistake. See Mantis 76.

Version 4.4

- Added support for the EOS 400D camera. New eos-dll redist files are included. See Mantis 71.
- Added RemoteGetNumberOfAvailableShots 18 method. See Mantis 58.
- Added DriveMode, AFMode, ColorSpace, WhiteBalanceShift remote parameters, mainly supported by EOS-DLL and newer EOS-cameras. See Mantis 59.
- Resolved problem to set AV/TV values in newer PowerShot cameras (e.g. S3IS, A620 and S80). See Mantis 57.
- Added the CacheRemoteParamDir parameter in the RemoteStart 30 method, making it possible to cache supported remote parameters. These cached remote parameters can then be used instead of probing the camera at each start-up, decreasing the time when starting remote mode to nearly nothing compared with up to 10 seconds. See Mantis 61.
- Added information in this manual about that PowerShot cameras always starts in 'Auto' mode. The camera mode set by the knobs on the camera does not have any affect, instead camera mode must always be set by software.
- Added support for the DriveMode (10 and 2 second self timer) remote parameters for newer PowerShot cameras (e.g. S3IS, A620 and S80). See Mantis 63.
- Added support for the AF Focusing Mode remote parameter for newer PowerShot cameras (e.g. S3IS, A620 and S80). See Mantis 64.
- Added support for the AF Assist Light remote parameter for newer PowerShot cameras (e.g. S3IS, A620 and S80). See Mantis 65.
- Added support for reading body id for newer PowerShot cameras (e.g. S3IS, A620 and S80). See Mantis 66.
- Temporary files and intermediate files are now stored in users temporary directory instead of program directory, which could be write protected. See Mantis 60.
- Resolved problem "Not possible to disconnect/connect with EOS 30D". See Mantis 69.
- Memory leakage tests performed. A few small issues were resolved. See Mantis 49.
- Source code quality checked for e.g. name clashes of Delphi identifiers (e.g. Time, Name). See Mantis 70.
- Resolved problem "TCamRemote precompiled version will not compile if TRegWare component is installed". See Mantis 52.
- Resolved problem "TCamRemote sometimes raises an exception when reading firmware version from EOS-cameras". See Mantis 72.

- Resolved problem "Error when setting time (at daylight time) in the camera". See Mantis 35.
- Added the RemoteLoadCameraRemoteParams 23 and RemoteSaveCameraRemoteParams 23 methods enabling storing and retrieving of camera remote parameters to file. See Mantis 74.

Version 4.3

- Solved problem with support in EOS-DLL for the 20D and 350D cameras, which caused TCamRemote to freeze when starting remote operations. See Mantis 54.
- Resolved problem "Exception is raised in the EOS-DLL Connect method In Delphi 6 and Delphi 2006, but not Delphi 7". See Mantis 56.
- Added PictureStyle remote parameter, supported by newer EOS cameras (e.g. 5D and 30D). See Mantis 37.
- Added support for taking RAW+JPEG pictures, instead of only RAW. Added CompQualityPic2RAW and ImageSizePic2RAW remote parameters. See Mantis 36.
- Added GetFirmwareVersion method. See Mantis 55.

Version 4.2

- Updated all template application to guarantee that the CloseCameraEnumeration 4 method is called if a call to CloseCameraEnumeration 4 has been done. If CloseCameraEnumeration 4 not is called the application will terminate with a crash. See Mantis 51.

Version 4.1

- Resolved problem "Not possible to get any events when a picture has been taken with a Powershot camera". See Mantis 48.

Version 4.0

- Added new EOS-DLL interface and changed the old interface to EOS-OLD-DLL. The new EOS-DLL interface adds support for several new EOS digital cameras (e.g. EOS 30D). See Mantis 30.
- Added support for the PowerShot S3-IS camera. See Mantis 46.
- The EOS-DLL (but not EOS-OLD-DLL) interface supports new RAW development function. FileFormatJPEG is added to FileFormatType 9 making it possible to develop RAW-pictures to JPEG. Added ICCProfileFileName and JPEGQuality parameter to the DevelopRAWPicture 9 method, making it possible to assign an ICC-profile for the developed picture and to setthe JPEG quality when developing JPEG-pictures. See Mantis 45.
- GetBodyID 10 method added. See Mantis 43.
- Added PictureType parameter in the RemoteGetPicture 18 method.
- NotifyRemoteEventValidType changed in OnRemoteEvent 40.
- Added events in the Event parameter in the OnEvent 38 callback method (added elements in EventEnumType 44).
- Changed the RemoteTakePicture 36 method from function to procedure.
- Changes in the returned list of pictures in the OpenCameraCollection 12 method, regarding RAW pictures taken with RAW+JPEG.

Version 3.2

- Resolved problems when compiling TCamRemote source code with either EOSDLL or PSDLL (not both at once). See Mantis 25.
- Increased the trial period to 60 days and removed the time tampering check. See Mantis 28.
- Time trial functions are enabled/disabled using the new REGISTRATION_NEEDED compiler switch. See Mantis 26.
- ISO remote parameter is possible to set in 1/3 step. See Mantis 27.
- Added camera card format functions using method FormatCameraCard 11. See Mantis 29.
- Added RemoteAFLock 17 method to set or unset camera AF lock during remote operations. See Mantis 24.

Version 3.1

- Changed the trial function. The developer are granted a 30 days trial period from the time where TCamRemote is used for the first time. This applies also to a user of a program which includes a trial version of TCamRemote. It is possible to enter registration data in the RegCode and RegName VCL parameters and register TCamRemote.
- Fixed a potential problem when enumerating pictures using method OpenCameraCollection 12.

- RAWJpegExists added to ImageDataType.
- Added functions to get the embedded JPEG file stored in a RAW-picture using method GetPicture 10.
- Added functions to develop an embedded JPEG file or a thumbnail stored from a RAW-picture using method DevelopRAWPicture 9.

Version 3.0

- Corrected problem with remote strobe handling, specially when handling Ixus cameras.
- Corrected problem when remotely taking pictures that only are stored on EOS camera memory.
- Improved setting of remote parameters for EOS cameras. TCamRemote tries to set each remote parameter in up to five seconds, if the camera is busy doing something else.
- Improved probing speed.
- A string list of connected cameras is returned when enumerating connecting cameras using the OpenCameraEnumeration 14 method.
- The first camera has the number 0 when calling the Connect 4 method.
- Added feedback of battery status when calling the Connect 4 method.
- Added SetTimeInCamera 37 method to make it possible to set date and time to a connected camera.
- Added parameter GetThumbnails to the OpenCameraCollection 12.
- Fixed a potential problem with threads in the RemoteTemplate application, when receiving viewfinder pictures from the camera. The VCLs must be updated within the main thread.
- Added RAW-development methods (OpenRAWObject 15, CloseRAWObject 4, GetRAWDevelopmentParameters 11, SetRAWDevelopmentParameters 37, DevelopRAWPicture 9 and OnRawDevelopEvent 39) and events making it possible to convert RAW-pictures from EOS-cameras to 8-bits or 16-bits TIFF and adjust some development parameters (e.g. whitebalance setting).
- Shutter and aperture values are possible to set in 1/3 step interval instead of 1 step interval, see RemoteFormatAVType and RemoteFormatTVType in CamDefines.pas.
- Added SetOwnerName 37 method.
- Added bulb shutter speed.
- Added support for remote handling of S80 and A620. It is however not possible to get pictures from the camera memory for these cameras.

Version 2.0

- Support for Delphi 2006 added.
- Support to handle multiple cameras at once, if the cameras are of different models (e.g. a PowerShot S40 a PowerShot S70 and a EOS 10D) added.
- Additional EOS cameras supported (EOS 5D and EOS-1D Mark II N).
- Corrected problems related to set/get ISO values to/from the camera.
- Parameter WhiteBalanceKelvin added to the RemoteReleaseParametersType 45 structure. The parameters is used to set the Kelvin degrees, when shooting with manual whitebalance settings.

Version 1.4

- New event OnRemoteProbeParamEvent 42.
- Event OnRemoteEvent 40 changed to handle EOS cameras better. RemoteEventCallbackReleaseImageReady added in RemoteEventCallbackType and parameter EventData added.
- The parameter SyncMode is removed from the RemoteTakePicture 36 method, since that mode not is supported by all EOS cameras (newer than EOS 20D). All cameras do however support the asynchronized file transfer mode.
- Problems to support newer EOS cameras including EOS 20D solved.
- Template application updated to support taking picture with the shutter while downloading a file from the camera at the same time for EOS cameras.

Version 1.3.

- New PowerShot and EOS cameras supported.
- Parameter ProbeRemoteParameters added to the RemoteStart 30 method.
- The following types have changed (more remote parameters added, but some items in the types

may have switched order). RemoteFormatQualityType, RemoteFormatSizeType, RemoteFormatShootingModeType, RemoteFormatWBType, RemoteFormatAFDistType. The template application has been updated to use those new remote parameters.

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.

2.5 Installation ActiveX

Setup TCamRemote ActiveX

ActiveX Component (stored in the OCX directory):
CamRemoteActiveXXControl1.ocx

The ocx-file must be registered using regsvr32.exe. It is very important to copy all files to the target computer, before registering these files. Most setup toolkits have an option to automate this task. The ocx-file must increase their reference counts in HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\SharedDLLs. Most setup toolkits have options to perform this step automatically.

Runtime files (stored in the redist directory):
Press this link for more information. 2

Remove the TCamRemote icon from the form during run-time:
Set Visible property to false during runtime of an application to hide the TCamRemote ActiveX design icon on window forms.

2.6 Known problems

The ActiveX version includes PS-DLL interface for PowerShot support and EOS-DLL interface for EOS support. The EOS-OLD-DLL interface is not supported by the ActiveX version. The known problems stated below is written for the Delphi VCL version, but is also valid in most cases for the ActiveX version. The data type may differ though.

- In the EOS-DLL interface it is not possible to get thumbnails from remotely taken pictures. `EventData.TypeOfPicture` parameter will only be set to `TypeOfPicturePicture` (parameter `Event` set to `RemoteEventCallbackReleaseImageReady`) when the TCamRemote calls the OnRemoteEvent 40 callback method. See Mantis 44.
- In the EOS-DLL interface it is only possible to get thumbnails and EXIF data from JPEG files or JPEG files embedded in RAW files stored on the camera card. For RAW-files EXIF and thumbnails will be set to nil. The thumbnail for JPEG pictures includes correct EXIF information, but the preview JPEG picture is incorrect and includes only a grey picture. See Mantis 42.
- The following cameras do only support remote operations mode (not possible to download pictures stored in the camera card or to format the camera card): PowerShot S80, A620, PowerShot S3 IS. See Mantis 47.
- The thumbnail picture is corrupt for pictures taken remote for the PowerShot S3IS camera. See Mantis 73.
- It seems not to be possible to connect and remote handle two EOS cameras of different models, using the EOS-DLL interface. This is however possible in the EOS-OLD-DLL interface.
- Set Visible property to false during runtime of an application to hide the TCamRemote ActiveX design icon on window forms.

2.7 Support

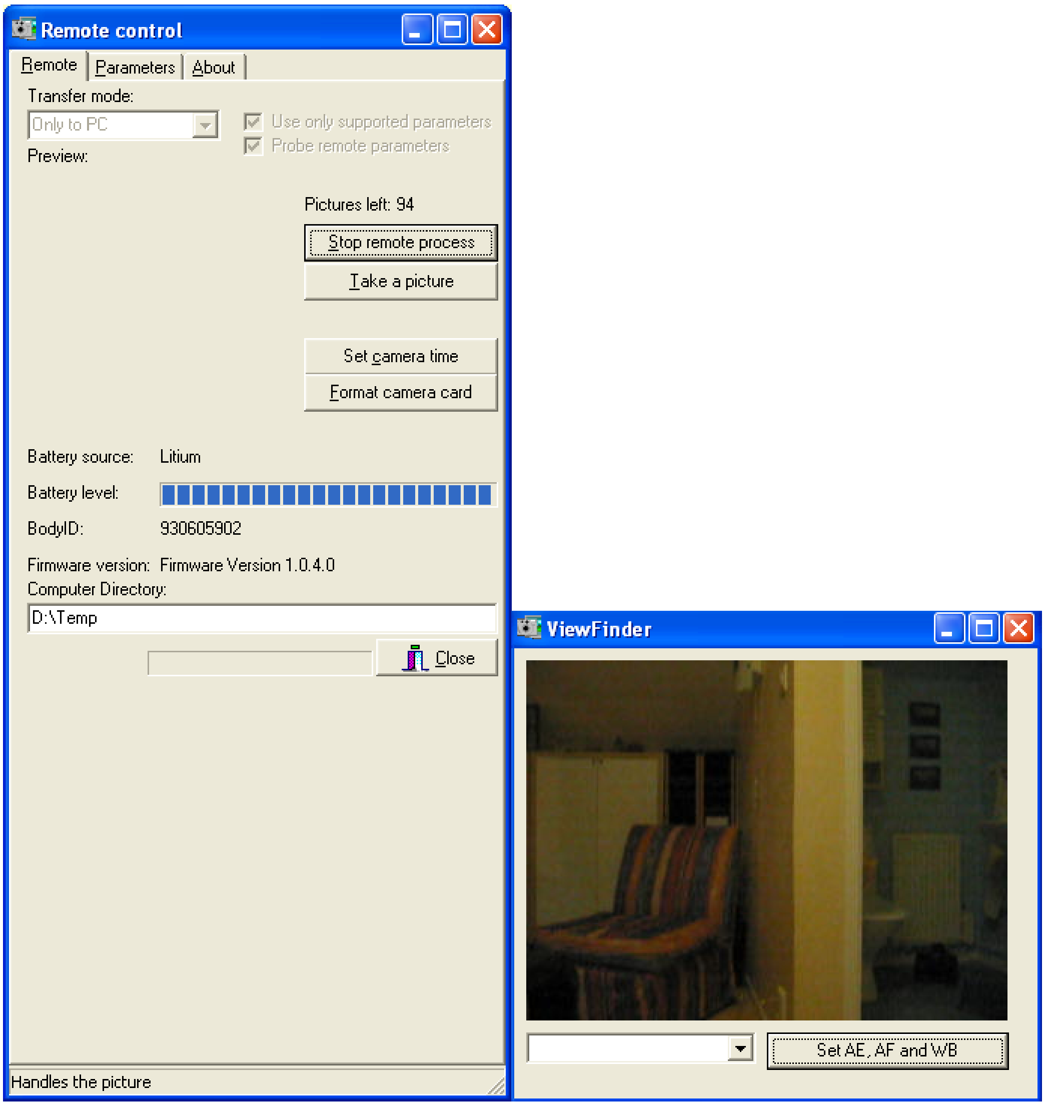
TCamRemote uses a Mantis bug tracking system to handle problem reports and change requests. User feedback on functions (bugs or requests for new functions) are possible for any user to create but also to follow up issues entered. Notes are used for feedback on all issues and is the most important way to communicate between the developer/support and the user. The Mantis system is available at: http://alkenius.no-ip.org/mantisbt/

New users of TCamRemote can register for an account at no cost. Each account will then be processed by the owner of TCamRemote before the account is enabled.

2.8 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 "Use only supported parameters is checked, only supported remote parameters probed when calling the RemoteStart 30 method is showed in the Parameters tab. When not checked all remote parameters are showed in the Parameters tab.



Transfer mode: Only to PC

☑ Use only supported parameters

☑ Probe remote parameters

Pictures left: 94

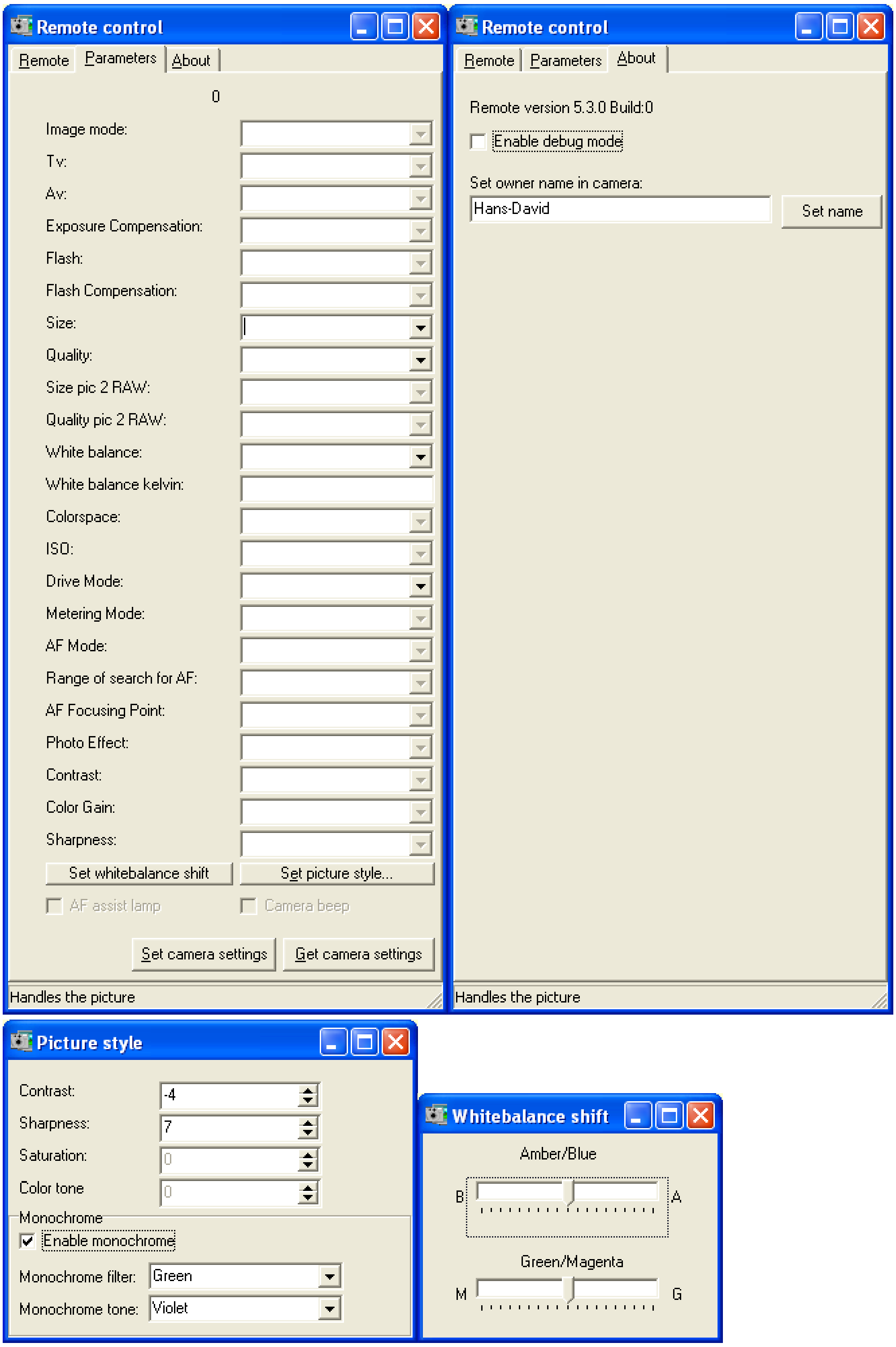
Battery source: Litium

BodyID: 930605902

Firmware version: Firmware Version 1.0.4.0

Computer Directory: D:\Temp

Handles the picture



Remote version 5.3.0 Build:0

☐ Enable debug mode

Set owner name in camera: Hans-David

Contrast: -4

Sharpness: 7

Saturation: 0

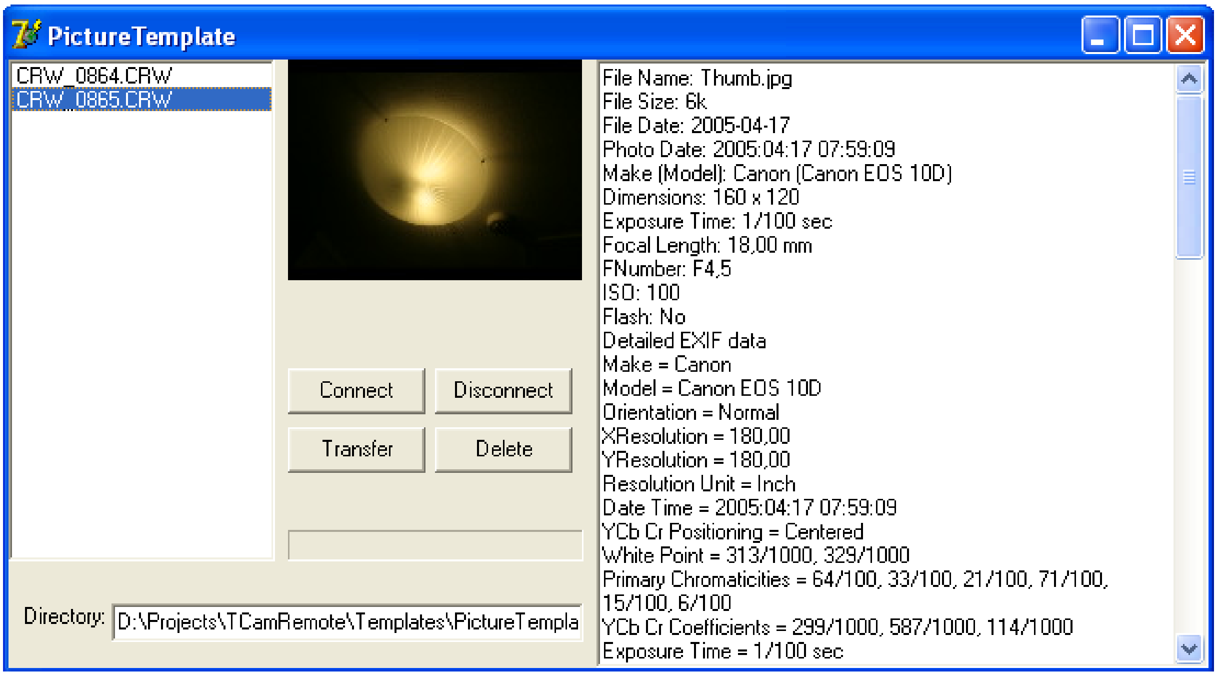
Color tone 0

☑ Enable monochrome

Monochrome filter: Green

Monochrome tone: Violet

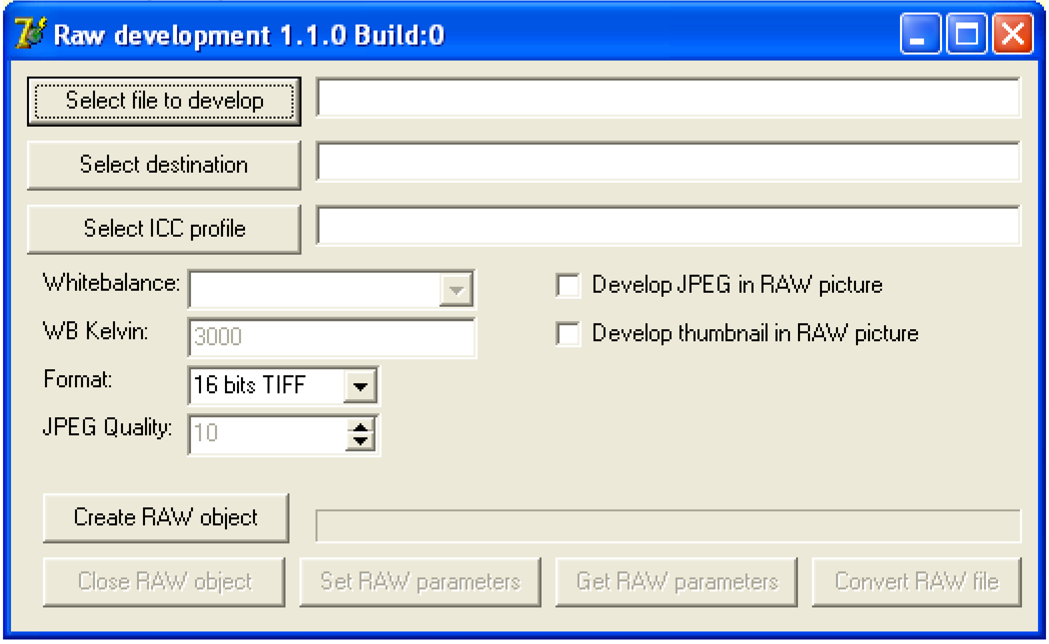
PictureTemplate:
The PictureTemplate application lists pictures stored on the camera when the Connect button is pressed. Selected picture can be transferred to the computer pressing the Transfer button and deleted pressing the Delete button.



Directory: D:\Projects\TCamRemote\Templates\PictureTempla

```
File Name: Thumb.jpg
File Size: 6k
File Date: 2005-04-17
Photo Date: 2005:04:17 07:59:09
Make (Model): Canon (Canon EOS 10D)
Dimensions: 160 x 120
Exposure Time: 1/100 sec
Focal Length: 18,00 mm
FNumber: F4,5
ISO: 100
Flash: No
Detailed EXIF data
Make = Canon
Model = Canon EOS 10D
Orientation = Normal
XResolution = 180,00
YResolution = 180,00
Resolution Unit = Inch
Date Time = 2005:04:17 07:59:09
YCb Cr Positioning = Centered
White Point = 313/1000, 329/1000
Primary Chromaticities = 64/100, 33/100, 21/100, 71/100,
15/100, 6/100
YCb Cr Coefficients = 299/1000, 587/1000, 114/1000
Exposure Time = 1/100 sec
```

RawDevelopTemplate:



WB Kelvin: 3000
Format: 16 bits TIFF
JPEG Quality: 10

2.9 RAW development

It is possible to develop RAW pictures (both CRW and CR2 are supported) to either 8 or 16 bits TIFF pictures, but only for RAW pictures taken by EOS cameras.
The TCamRemote component can also develop RAW-pictures to JPEG pictures and use ICC-profiles during development.

Currently only one development parameter (whitebalance) are able to be set. In the future more parameters will be added. The development parameters is used to override the parameters set by the camera when the picture was taken.

The following methods are used for RAW development:

- OpenRAWObject 15
- CloseRAWObject 4

- GetRAWDevelopmentParameters 11
- SetRAWDevelopmentParameters 37
- DevelopRAWPicture 9
- OnRawDevelopEvent 39

Test the RawDevelop template 10 to get a hint of what is possible.

2.10 Handle several cameras

It is possible to handle multiple cameras at once using TCamRemote. One at a time or multiple cameras at once. However there is one big drawback. It is **only** possible to handle several cameras of different models at once. There is no problem to handle several cameras of the same model, but it is only possible to remotely handle one at a time.

This means that it is **not** possible to use 2 PowerShot S70 at once, but it is possible to use a PowerShot S40 a PowerShot S70 and an EOS 20D at once. There are no restrictions on how many cameras that can be handled at once.

To handle multiple cameras at once must the application be threaded where each thread includes a TCamRemote object. It is not recommended to use TCamRemote visual objects (available on forms). Instead create a private TCamRemote object in each thread. The thread will be responsible to create and destroy the TCamRemote object. It is possible to use callback event function normally available as events in a visual components, by creating the methods prototypes including code and assign the events to these procedure. Below a Delphi example from the VCL version is available.

```
//Event callback function for remote camera events
procedure TCamThread.RemoteEvent(Event     : RemoteEventCallbackType;
                                 EventData : AdditionNotifyRemoteEventType);
begin
  if (Event = RemoteEventCallbackReleaseComplete) then
  begin
    mPicturesToReceive := mPicturesToReceive + EventData.NumOfEvents;
  end;
  if (Event = RemoteEventCallbackReleaseImageReady) then
  begin
    mPicturesToReceive := mPicturesToReceive + 1;
  end;
end;

procedure TCamThread.Execute;
var p_command   : ^TCamCommand;
    time_string : string;
begin
  mCommand                           := TList.Create;
  mCamRemote                         := TCamRemote.Create(nil);
  mRemoteOn                          := false;
  mNewViewfinder                     := false;
  mViewfinderOn                      := false;
  mPicturesToReceive                 := 0;
  mViewfinderPic                     := TJpegImage.Create;
  //Assign events
  mCamRemote.OnViewfinderEvent       := ViewfinderEvent;
  mCamRemote.OnEvent                 := CamRemoteEvent;
  mCamRemote.OnRemoteGetPictureEvent := TakePictureEvent;
  mCamRemote.OnRemoteEvent           := RemoteEvent;
  try
    while (not terminated) do
    begin
      try
        if (mCommand.Count > 0) then
        begin
          p_command := mCommand[0];
          case p_command^ of
            CamOpenCameraEnumeration :
              begin
                 mCamerasConnected := mCamRemote.OpenCameraEnumeration;
                 Synchronize(UpdateConnectedCameras);
              end;
            CamConnect :
              begin
                mCameraInfo       := mCamRemote.Connect(mCameraToUse.CameraType,
                                                        mCameraToUse.CameraNumber);
                mCameraCapability := mCamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                                            ReleaseDataKindTakeOnlyPicture,
                                                            false);  //No probing
                mTurnViewfinderBtnOn := mCameraCapability.DoSupportViewfinder;
                Synchronize(UpdateViewfinderButtons);
                mRemoteOn := true;
              end;
            CamDisconnect :
              begin
                if (mViewfinderOn) then
                begin
                  mCamRemote.RemoteStopViewfinder;
                  mViewfinderOn := false;
                end;
                mCamRemote.RemoteEnd;
                mCamRemote.Disconnect;
                mTurnViewfinderBtnOn := false;
                Synchronize(UpdateViewfinderButtons);
                mRemoteOn := false;
              end;
            CamTakePicture :
              begin
                if ((mCameraCapability.ReqViewfinderOffWhenShooting) and (mViewfinderOn)) then
                begin
                  //Turn off viewfinder temporarily
                  mCamRemote.RemoteStopViewfinder;
                end;
                mCamRemote.RemoteTakePicture;
                if ((mCameraCapability.ReqViewfinderOffWhenShooting) and (mViewfinderOn)) then
                begin
                  //Turn on viewfinder
                  mCamRemote.RemoteStartViewfinder;
                end;
              end;
            CamStartViewfinder :
              begin
                mCamRemote.RemoteStartViewfinder;
                mViewfinderOn := true;
              end;
            CamStopViewfinder :
              begin
                 mCamRemote.RemoteStopViewfinder;
                 mViewfinderOn := false;
              end;
            CamCloseCameraEnumeration :
              begin
                 mCamRemote.CloseCameraEnumeration;
              end;
          end;
          mCommand.Delete(0);
        end;
        if (mNewViewfinder) then
        begin
          Synchronize(UpdateMainFormPicture);
        end;
        //Check if a picture is available
        if (mPicturesToReceive > 0) then
        begin
          DateTimeToString(time_string, 'yymmddhhnnss_zzz', Now);
          mCamRemote.RemoteGetPicture(time_string + '.jpg', TypeOfPicturePicture);
          mPicturesToReceive := mPicturesToReceive - 1;
        end;
        Sleep(100); //Let other threads execute
      except
        //Special exception handler
        HandleException;
      end;
    end; //While (not terminated) do
    if (mViewfinderOn) then
    begin
      mCamRemote.RemoteStopViewfinder;
      mViewfinderOn := false;
    end;
    if (mRemoteOn) then
    begin
      mCamRemote.RemoteEnd;
```

```
      mCamRemote.Disconnect;
      mTurnViewfinderBtnOn := false;
      Synchronize(UpdateViewfinderButtons);
      mRemoteOn := false;
    end;
  finally
    mCommand.Free;
    mCamRemote.Free;
  end;
end;
```

Part III

3 TCamRemote component

3.1 TCamRemote

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

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,
- develop RAW pictures to TIFF and JPEG.

3.2 TCamRemote runtime files

The TCamRemote uses DLLs to interface the PowerShot and EOS cameras and to develop RAW pictures. The DLLs are stored in the redist directory.
TCamRemote 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.

3.4 TCamRemote methods

3.4.1 List of methods in TCamRemote

CloseCameraCollection 3
CloseCameraEnumeration 4
CloseRAWObject 4
Connect 4
Connect_CameraModel 6
Connect_CameraModelName 7
Connect_OwnerName 7
Connect_Battery_BatterySource 7
Connect_Battery_BatteryStatus 8
DeletePicture 8
DevelopRAWPicture 9
Disconnect 9
GetBodyID 10
GetOwnerName 10
GetPicture 10
GetRAWDevelopmentParameters 11

FormatCameraCard 11
OpenCameraCollection 12
OpenCameraCollection_GetNrOfPictures 13
OpenCameraCollection_NameOfImage 13
OpenCameraCollection_GetThumbnail 13
OpenCameraEnumeration 14
OpenCameraEnumeration_GetPowerShotNr 14
OpenCameraEnumeration_GetEOSNr 15
OpenCameraEnumeration_GetModel 15
OpenRAWObject 15
OpenRAWObject_GetRemoteParamSupported 16
RemoteActivateViewfinderAuto 17
RemoteAFLock 17
RemoteEnd 18
RemoteGetNumberOfAvailableShot 18
RemoteGetPicture 18
RemoteGetRemoteParams 19
RemoteGetRemoteParams_PictureStyle 21
RemoteGetRemoteParams_WhitebalanceShift 21
RemoteGetZoomPos_CurrentZoomPos 22
RemoteGetZoomPos_MaxOpticalZoomPos 22
RemoteGetZoomPos_MaxZoomPos 23
RemoteLoadCameraRemoteParams 23
RemoteSaveCameraRemoteParams 23
RemoteSetRemoteParams 24
RemoteSetRemoteParams_ImageQuality 27
RemoteSetRemoteParams_PictureStyle 27
RemoteSetRemoteParams_WhitebalanceShift 28
RemoteSetViewfinderOutput 28
RemoteSetZoomPos 29
RemoteStart 30
RemoteStart_DoSupportZoom 31
RemoteStart_DoSupportShootingPara 31
RemoteStart_DoSupportViewfinder 32
RemoteStart_ReqViewfinderOffWhenShooting 32
RemoteStart_DoSupportAfLockUnlock 32
RemoteStart_GetRemoteParamSupported 33
RemoteStartViewfinder 35
RemoteStopViewfinder 35
RemoteSupported 36
RemoteTakePicture 36
SetOwnerName 37
SetRAWDevelopmentParameters 37
SetTimeInCamera 37

3.4.2 TCamRemote.CloseCameraCollection

Closes the volume for pictures on the camera and frees the data structure returned from the OpenCameraCollection 12 method.

Syntax:

```
procedure CloseCameraCollection;
```

Prerequisite:

The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 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.

If any errors occurs an ECamException 2 exception will be raised.

3.4.2.1 CloseCameraCollection example

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

3.4.3 TCamRemote.CloseCameraEnumeration

Closes communication to enumerated connected cameras.

Syntax:

```
procedure CloseCameraEnumeration;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
The CloseCameraEnumeration method closes the communication to connected cameras and frees resources used by the TCamRemote components.

If any errors occurs an ECamException 2 exception will be raised.

3.4.4 TCamRemote.CloseRAWObject

Closes the RAW object.

Syntax:

```
procedure CloseRAWObject;
```

Prerequisite:
The RAW object has been created using the OpenRAWObject 15 method.

Description:
The CloseRAWObject method closes RAW object and frees the resources used by the TCamRemote component for RAW development.

If any errors occurs an ECamException 2 exception will be raised.

3.4.5 TCamRemote.Connect

Connects to a PowerShot or EOS camera.

Syntax:

```
procedure Connect(CameraType   : CamModType 44 ;
                  CameraNumber : integer);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
The Connect method tries to establish a connection to a PowerShot or EOS camera. The connection to the camera will remain until the Disconnect 9 method is called or the camera is disconnected from

the computer (the USB cable is ejected or battery level very critical).
After a successful connection

- the OnEvent 38 event is called, when a camera event has occurred.

Use the Connect_XXX methods to get information about the connected camera.

Parameter:
- CameraType: Sets the camera type (PowerShot or EOS) to connect to.
- CameraNumber: The number of the camera to connect to. The first camera connected has the number 0. The OpenCameraEnumeration_XXX methods can be used to get information about the connected camera before connecting to one.

Please note the following:

- It is possible to connect to several PowerShot and EOS cameras at once (using threads and several TCamRemote objects), if they are of different models (e.g. a PowerShot S40 and a S70). It is not possible to connect to several cameras of the same model at once. For more information see chapter how to handle several cameras at once 13.

See also:
Connect_CameraModel 6, Connect_CameraModelName 7, Connect_OwnerName 7, Connect_Battery_BatterySource 7, Connect_Battery_BatteryStatus 8

If any errors occurs an ECamException 2 exception will be raised.

3.4.5.1 Connect example

```
//Procedure to connect to a camera
procedure TFormRemote.ConnectToCamera;
var i : integer;
begin
  //See if already conected
  if (not mConnToCamera) then
  begin
    try
      CamRemoteActiveX.OpenCameraEnumeration;
      if ((CamRemoteActiveX.OpenCameraEnumeration_GetPowerShotNr = 0) and
          (CamRemoteActiveX.OpenCameraEnumeration_GetEOSNr = 0)) then
      begin
        raise ECamException.Create('No camera is connected to the computer');
      end;
      FormSelectCamera.ModalResult := mrOK;
      FormSelectCamera.ComboBoxCameras.Items.Clear;
      //Are there any PowerShot cameras connected?
      if (CamRemoteActiveX.OpenCameraEnumeration_GetPowerShotNr > 0) then
      begin
        if (CamRemoteActiveX.OpenCameraEnumeration_GetPowerShotNr > 1) then
        begin
          //Select which camera to connect to
          for i := 0 to (CamRemoteActiveX.OpenCameraEnumeration_GetPowerShotNr - 1) do
          begin
            FormSelectCamera.ComboBoxCameras.Items.Add(CamRemoteActiveX.OpenCameraEnumeration_
GetModel(Ord(CamModPowerShot), i));
          end;
          FormSelectCamera.ComboBoxCameras.ItemIndex := 0;
          if (FormSelectCamera.ShowModal = mrOK) then
          begin
            CamRemoteActiveX.Connect(Ord(CamModPowerShot),
                                     FormSelectCamera.ComboBoxCameras.ItemIndex);
          end;
        end else begin
          //Only one PowerShot camera connected. Select that camera
          CamRemoteActiveX.Connect(Ord(CamModPowerShot), 0);
        end;
      end else begin
        //No PowerShot cameras. Connect to EOS camera
        if (CamRemoteActiveX.OpenCameraEnumeration_GetEOSNr > 1) then
        begin
          //Select which camera to connect to
          for i := 0 to (CamRemoteActiveX.OpenCameraEnumeration_GetEOSNr - 1) do
          begin
            FormSelectCamera.ComboBoxCameras.Items.Add(CamRemoteActiveX.OpenCameraEnumeration_
GetModel(Ord(CamModEOS), i));
```

```
          end;
          FormSelectCamera.ComboBoxCameras.ItemIndex := 0;
          if (FormSelectCamera.ShowModal = mrOK) then
          begin
            CamRemoteActiveX.Connect(Ord(CamModEOS),
                                     FormSelectCamera.ComboBoxCameras.ItemIndex);
          end;
        end else begin
          //Only one EOS camera connected. Select that camera
            CamRemoteActiveX.Connect(Ord(CamModEOS), 0);
        end;
      end;
      if (FormSelectCamera.ModalResult <> mrOK) then
      begin
        Exit;
      end;
      mCameraInfo.CameraModel :=
        CamModType(CamRemoteActiveX.Connect_CameraModel);
      mCameraInfo.CameraModelName :=
        CamRemoteActiveX.Connect_CameraModelName;
      mCameraInfo.OwnerName :=
        CamRemoteActiveX.Connect_OwnerName;
      mCameraInfo.Battery.BatterySource :=
        BatterySourceType(CamRemoteActiveX.Connect_Battery_BatterySource);
      mCameraInfo.Battery.BatteryStatus :=
        BatteryStatusType(CamRemoteActiveX.Connect_Battery_BatteryStatus);
      //Update battery status
      LabelBatterySource.Caption := '-';
      case mCameraInfo.Battery.BatterySource of
        BatterySourceAC     : LabelBatterySource.Caption := 'AC';
        BatterySourceLitium : LabelBatterySource.Caption := 'Litium';
        BatterySourceNiMH   : LabelBatterySource.Caption := 'NiMH';
        BatterySourceNiCD   : LabelBatterySource.Caption := 'NiCD';
        BatterySourceAlMN   : LabelBatterySource.Caption := 'AlMN';
      end;
      ProgressBarBattery.Position := 1;
      case mCameraInfo.Battery.BatteryStatus of
        BatteryStatusNormal    : ProgressBarBattery.Position := 5;
        BatteryStatusWeak      : ProgressBarBattery.Position := 3;
        BatteryStatusSafetyLow : ProgressBarBattery.Position := 2;
      end;
      //Update BodyID
      LabelBodyIDValue.Caption := Format('%u', [CamRemoteActiveX.GetBodyID]);
      //Update owner name
      EditOwnerName.Text := mCameraInfo.OwnerName;

      StatusBar.SimpleText := 'Camera is connected';
      mConnToCamera := true;
      Log('Camera model='        + mCameraInfo.CameraModelName);
      Log('Camera name='         + mCameraInfo.OwnerName);
      Log('Camera connected');
    except
      on E:exception do
        begin
          log('Connect_to_powershot. Exception=' + E.Message);
          StatusBar.SimpleText := 'Connection to camera failed';
          mConnToCamera        := false;
          raise;
        end;
    end;
  end;
end;
```

3.4.6 TCamRemote.Connect_CameraModel

Returns the connected camera type (e.g. PowerShot or EOS).

Syntax:

```
function  Connect_CameraModel : CamModType 44;
```

Prerequisite:

The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:

-

See also:
Connect 4, Connect_CameraModelName 7, Connect_OwnerName 7, Connect_Battery_BatterySource 7, Connect_Battery_BatteryStatus 8

If any errors occurs an ECamException 2 exception will be raised.

3.4.7 TCamRemote.Connect_CameraModelName

Returns the connected camera model name (e.g. "PowerShot S70").

Syntax:
```
function  Connect_CameraModelName : string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
-

See also:
Connect 4, Connect_CameraModel 6, Connect_OwnerName 7, Connect_Battery_BatterySource 7, Connect_Battery_BatteryStatus 8

If any errors occurs an ECamException 2 exception will be raised.

3.4.8 TCamRemote.Connect_OwnerName

Returns the connected camera owner name.

Syntax:
```
function  Connect_OwnerName: string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
Returns the connected camera owner name. The owner name can be set in the camera by using the SetOwnerName 37 method.

See also:
Connect 4, Connect_CameraModel 6, Connect_CameraModelName 7, Connect_Battery_BatterySource 7, Connect_Battery_BatteryStatus 8

If any errors occurs an ECamException 2 exception will be raised.

3.4.9 TCamRemote.Connect_Battery_BatterySource

Returns the connected camera battery type (e.g. NiMH and Lithium).

Syntax:
```
function  Connect_Battery_BatterySource : BatterySourceType 44;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
-

See also:
Connect 4, Connect_CameraModel 6, Connect_CameraModelName 7, Connect_OwnerName 7, Connect_Battery_BatteryStatus 8

If any errors occurs an ECamException 2 exception will be raised.

3.4.10 TCamRemote.Connect_Battery_BatteryStatus

Returns the connected camera battery status (e.g. if the power in the battery is normal or weak).

Syntax:

```
function  Connect_Battery_BatteryStatus : BatteryStatusType 44;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
-

See also:
Connect 4, Connect_CameraModel 6, Connect_CameraModelName 7, Connect_OwnerName 7, Connect_Battery_BatterySource 7

If any errors occurs an ECamException 2 exception will be raised.

3.4.11 TCamRemote.DeletePicture

Deletes a picture in the camera.

Syntax:

```
procedure DeletePicture(NrInList : integer);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 method, to list pictures on the camera.

Description:
The DeletePicture method deletes a picture in the camera.

Parameter:
- NrInList: The array element in ImageList that is to be deleted in the camera.

If any errors occurs an ECamException 2 exception will be raised.

3.4.11.1 DeletePicture example

```
procedure TFormMain.ButtonDeleteClick(Sender: TObject);
begin
  if (ListBox.ItemIndex <> -1) then
  begin
    CamRemoteActiveX.DeletePicture(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 CamRemoteActiveX.OpenCameraCollection_GetNrOfPictures - 1 do
  begin
    ListBox.Items.Add(CamRemoteActiveX.OpenCameraCollection_NameOfImage(i))
  end;
end;
```

3.4.12 TCamRemote.DevelopRAWPicture

Develops a RAW object to a TIFF or JPEG file.

Syntax:

```
procedure DevelopRAWPicture(OutFileName        : string;
                            FileFormat         : FileFormatType;
                            BitsPerPixel       : BitsPerPixelType;
                            ICCProfileFileName : string;
                            JPEGQuality        : integer);

type  FileFormatType = (FileFormatTIFF,
                        FileFormatThumbnail,
                        FileFormatJPEGInRaw,
                        FileFormatJPEG);

type BitsPerPixelType = (BitsPerPixel24Bits,
                         BitsPerPixel48Bits);
```

Prerequisite:
The RAW object has successfully been created using the OpenRAWObject 15 method.

Description:
The DevelopRAWPicture method creates either

- a TIFF file in 8 or 16 bits colours.
- a JPEG picture.

The filename of the file is set by the OutFileName parameter.
The OnRawDevelopEvent 39 is **not** used during development of the destination file.
An ICC-profile may be assigned to be used when developing the RAW-file to the destination file.
EXIF-information is stored in the developed file, including information about the optional ICC-profile.

Parameter:
- OutFileName: The name of the TIFF file to create.
- FileFormat: The requested file to develop. Either a TIFF, a thumbnail or the JPEG embedded in the RAW-picture.
- BitsPerPixel: The number of bits per colour per pixel. Only valid when FileFormat is set to FileFormatTIFF.
- ICCProfileFileName: The filename of the ICC-profile that shall be used when creating the developed file. If no ICC-profile is wanted, set the parameter to ''. If no ICC-profile is set, the sRGB profile will be used.
- JPEGQuality: Sets the JPEG quality for the developed pictures. Valid values are 1 (lowest quality) to 10 (highest quality). Is only valid if the FileFormat parameter is set to FileFormatJPEG.

If any errors occurs an ECamException 2 exception will be raised.

3.4.13 TCamRemote.Disconnect

Disconnects from the connected camera.

Syntax:

```
procedure Disconnect;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 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 38 event will not be called anymore.

If any errors occurs an ECamException 2 exception will be raised.

3.4.14 TCamRemote.GetBodyID

Returns the body id (camera serial number) stored in the camera.

Syntax:

```
function GetBodyID : Cardinal;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

Description:
Returns the body id (camera serial number) stored in the camera. If no body id can be read, 0 is returned from this method.

If any errors occurs an ECamException 2 exception will be raised.

3.4.15 TCamRemote.GetOwnerName

Returns the owner name stored in the camera.

Syntax:

```
function GetOwnerName: string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

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

If any errors occurs an ECamException 2 exception will be raised.

3.4.16 TCamRemote.GetPicture

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

Syntax:

```
procedure GetPicture(NrInList : integer;
                     GetJPEGInRaw : boolean;
                     FileName : string);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 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 39 is used during transfer.

Parameter:
- NrInList: The array element in ImageList (received from method OpenCameraCollection 12) that is to be copied to the computer.
- **GetJPEGInRaw:** If set to enable GetPicture will download the embedded JPEG-picture from a RAW-picture stored in the camera. If not set the JPEG or RAW-picture itself will be downloaded from the camera. This parameter is not currently used,
- FileName: The filename for the copied file on the computer.

If any errors occurs an ECamException 2 exception will be raised.

3.4.16.1 GetPicture example

```
procedure TFormMain.ButtonTransferClick(Sender: TObject);
begin
  if (ListBox.ItemIndex <> -1) then
  begin
    //Get JPEG in RAW picture.
    if (CamRemoteActiveX.OpenCameraCollection_RAWJpegExists(ListBox.ItemIndex)) then
    begin
      CamRemoteActiveX.GetPicture(ListBox.ItemIndex,
                                  true,
                                  ChangeFileExt(EditPCDir.Text + '\' +
ListBox.Items[ListBox.ItemIndex],
                                                '.jpg'));
    end;
    CamRemoteActiveX.GetPicture(ListBox.ItemIndex,
                                false,
                                EditPCDir.Text + '\' + ListBox.Items[ListBox.ItemIndex]);
  end;
  ProgressBar.Position := 0;
end;

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

3.4.17 TCamRemote.GetRAWDevelopmentParameters

Gets the current RAW development parameters for the RAW object.

Syntax:
`function GetRAWDevelopmentParameters(RAWParamKind : TRAWParamType 50) : integer;`

Prerequisite:
The RAW object has successfully been created using the OpenRAWObject 15 method.

Description:
This method gets the value for the RAW development parameter set in the RAWParamkind parameter. The returned value shall be interpreted as follows:

RAWParamKind:	Interpret as
RAWParamWhiteBalanceSetting	RemoteFormatWBType 45
RAWParamWhiteBalanceKelvin	Kelvin degrees.

Parameter:
- RAWParamKind: The RAW development parameter to get value for.

If any errors occurs an ECamException 2 exception will be raised.

3.4.18 TCamRemote.FormatCameraCard

Formats the camera memory card (e.g. CF- or SD-card).

Syntax:
`procedure FormatCameraCard;`

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

Description:
The camera card (CF- or SD-card) will be formatted. After the card is formatted it is empty and is immediately ready to be used, even if the camera is in remote mode. It is **not** necessary to disconnect and the reconnect to the camera to store new pictures on the card.

If any errors occurs an ECamException 2 exception will be raised.

3.4.19 TCamRemote.OpenCameraCollection

Opens a picture volume on the camera and enumerates the pictures stored on the camera memory.

Syntax:

```
procedure OpenCameraCollection(TempDir       : string;
                               GetThumbnails : boolean);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

Description:
OpenCameraCollection opens a volume on the camera and searches for pictures stored on the camera.

Please note the following:
- The EOS cameras are able to take a picture including both a RAW pictures with embedded JPEG pictures, e.g. "RAW + JPEG quality medium size large". The RAW and JPEG picture are separate image items in the picture volume, with the same name but different file extension (e.g. CR2 for the RAW and .JPG for the embedded JPEG picture).

Parameter:
- TempDir: A temporary directory where TCamRemote can create temporary thumbnail files, when pictures are searched in the camera.
- GetThumbnails: If set thumbnails data will be included in the returned list of objects. Getting thumbnails requires much time and may be time consuming.

See also:
OpenCameraCollection_GetNrOfPictures 13, OpenCameraCollection_NameOfImage 13, OpenCameraCollection_GetThumbnail 13

If any errors occurs an ECamException 2 exception will be raised.

3.4.19.1 OpenCameraCollection example

```
  //List all pictures on the camera
  CamRemoteActiveX.OpenCameraCollection(ExtractFileDir(ParamStr(0)),
                                        CheckBoxThumbnails.Checked);
  UpdateListBox;

procedure TFormMain.UpdateListBox;
var i : integer;
begin
  //Update the list of pictures on the form
  ListBox.Items.Clear;
  for i := 0 to CamRemoteActiveX.OpenCameraCollection_GetNrOfPictures - 1 do
  begin
    ListBox.Items.Add(CamRemoteActiveX.OpenCameraCollection_NameOfImage(i))
  end;
end;

//Procedure called when user clicks on a picture filename on the ListBox updated above.
procedure TFormMain.ListBoxClick(Sender: TObject);
var thumb_file_name : string;
begin
  //Print EXIF data for RAW-pictures to a RichEdit VCL.
  RichEdit.Lines.Clear;
  if ((ListBox.ItemIndex <> -1) and (CheckBoxThumbnails.Checked)) then
  begin
    thumb_file_name := GetEnvironmentVariable('Temp') + '\Thumb.jpg';
    CamRemoteActiveX.OpenCameraCollection_GetThumbnail(ListBox.ItemIndex, thumb_file_name);
    //Update thumbnail on the form
    ImageThumbnail.Picture.LoadFromFile(thumb_file_name);
  end;
end;
```

3.4.20 TCamRemote.OpenCameraCollection_GetNrOfPictures

Returns the number of pictures stored in the picture volume.

Syntax:

```
function  OpenCameraCollection_GetNrOfPictures : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 method, to list pictures on the camera.

Description:
The number of pictures stored on the picture volume, opened by the OpenCameraCollection 12 method, is returned.

See also:
OpenCameraCollection 12, OpenCameraCollection_NameOfImage 13, OpenCameraCollection_GetThumbnail 13

If any errors occurs an ECamException 2 exception will be raised.

3.4.21 TCamRemote.OpenCameraCollection_NameOfImage

Returns the filename of a picture stored in the picture volume.

Syntax:

```
function  OpenCameraCollection_NameOfImage(PictureNumber : integer) : string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 method, to list pictures on the camera.

Description:
The name of the picture with index set in the PictureNumber parameters is returned. The picture is stored in a picture volume opened by the OpenCameraCollection 12 method. The number of stored pictures is received using the OpenCameraCollection_GetNrOfPictures 13 method.

Parameter:
- PictureNumber: The index of the picture in the volume to get information from. The first index i 0.

See also:
OpenCameraCollection 12, OpenCameraCollection_GetNrOfPictures 13, OpenCameraCollection_GetThumbnail 13

If any errors occurs an ECamException 2 exception will be raised.

3.4.22 TCamRemote.OpenCameraCollection_GetThumbnail

Gets a thumbnail for a picture stored in the picture volume.

Syntax:

```
procedure OpenCameraCollection_GetThumbnail(PictureNumber : integer;
                                            FileName      : string);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 method, to list pictures on the camera.

Description:
This method gets a thumbnail for a picture which index is set by the PictureNumber parameter. The requested file name for the thumbnail is set in the FileName parameter. The picture is stored in a picture volume opened by the OpenCameraCollection 12 method. The number of stored pictures is received using the OpenCameraCollection_GetNrOfPictures 13 method.

Please note the following:

- The GetThumbnails parameter in the OpenCameraCollection 12 method must be set to true prior calling this method.

Parameter:
- PictureNumber: The index of the picture in the volume to get a thumbnail from. The first index i 0.
- Filename: The requested filename for the thumbnail.

See also:
OpenCameraCollection 12, OpenCameraCollection_GetNrOfPictures 13, OpenCameraCollection_NameOfImage 13

If any errors occurs an ECamException 2 exception will be raised.

3.4.23 TCamRemote.OpenCameraEnumeration

Enumerates connected cameras.

Syntax:
```
procedure OpenCameraEnumeration;
```

Prerequisite:
The runtime files 2 needs to be copied to the directory from which the application is executing.
The camera(s) needs to be connected to the computer and the connection must be active.

Description:
The OpenCameraEnumeration method scans the computer ports for connected cameras.
This method also initializes the TCamRemote component, and must therefore be called prior using any other method in the TCamRemote component.
Use the OpenCameraCollection_XXX methods to get the results from the scan of connected cameras.

Please note the following:

- If several TCamRemote objects are used, it is only required to call the OpenCameraEnumeration method from one of the TCamRemote objects to initialize the environment for all TCamRemote components.

See also:
OpenCameraEnumeration_GetPowerShotNr 14, OpenCameraEnumeration_GetEOSNr 15, OpenCameraEnumeration_GetModel 15

If any errors occurs an ECamException 2 exception will be raised.

3.4.24 TCamRemote.OpenCameraEnumeration_GetPowerShotNr

Returns the number of PowerShot cameras connected to the PC.

Syntax:
```
function  OpenCameraEnumeration_GetPowerShotNr : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
-

See also:
OpenCameraEnumeration 14, OpenCameraEnumeration_GetEOSNr 15,
OpenCameraEnumeration_GetModel 15

If any errors occurs an ECamException 2 exception will be raised.

3.4.25 TCamRemote.OpenCameraEnumeration_GetEOSNr

Returns the number of EOS cameras connected to the PC.

Syntax:
```
function  OpenCameraEnumeration_GetEOSNr : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
-

See also:
OpenCameraEnumeration 14, OpenCameraEnumeration_GetPowerShotNr 14,
OpenCameraEnumeration_GetModel 15

If any errors occurs an ECamException 2 exception will be raised.

3.4.26 TCamRemote.OpenCameraEnumeration_GetModel

Returns the camera model name of either a connected PowerShot or EOS camera.

Syntax:
```
function  OpenCameraEnumeration_GetModel(CameraType : CamModType 44;
                                         Nr         : integer) : string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.

Description:
This method returns the camera model name (e.g. "EOS 5D") of the camera defined by the CameraType and Nr parameters.

Parameter:
- CameraType: EOS or PowerShot camera model is wanted.
- Nr: .The number of the camera to get the model name from. The first camera has the number 0. The number of cameras connected is received by the OpenCameraCollection_GetPowerShotNr 14 and OpenCameraCollection_GetEOSNr 15

See also:
OpenCameraEnumeration 14, OpenCameraEnumeration_GetPowerShotNr 14,
OpenCameraEnumeration_GetEOSNr 15

If any errors occurs an ECamException 2 exception will be raised.

3.4.27 TCamRemote.OpenRAWObject

Creates a RAW object of a RAW file.

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

Prerequisite:
The runtime files 2 needs to be copied to the directory from which the application is executing.
The FileName must be a valid RAW file.

Description:

- The OpenRAWObject method creates a RAW object from the file defined in the FileName parameter. The RAW file is probed for valid RAW development parameters. Use the OpenRAWObject_GetRemoteParamSupported 16 to get the probed RAW development parameters.

Parameter:
- FileName: The filename of a valid RAW file.

See also:
OpenRAWObject_GetRemoteParamSupported 16

If any errors occurs an ECamException 2 exception will be raised.

3.4.28 TCamRemote.OpenRAWObject_GetRemoteParamSupported

Gets the RAW development parameters that are supported for the RAW object.

Syntax:

```
function  OpenRAWObject_GetRemoteParamSupported(RAWParamKind : TRAWParamType 50 ) : string;
```

Prerequisite:
The RAW object has successfully been created using the OpenRAWObject 15 method.

Description:
This method gets the RAW development parameters that are supported for the RAW object. These RAW development parameters are probed in the OpenRAWObject 15 method. One parameters at a time can be received, set by the RAWParamKind parameter. The returned value is a string which length is equal the data type listed below (e.g. RemoteFormatWBType 45 for RAWParamWhiteBalanceSetting). Each character in the string is either a '0' or '1'. '0' is interpreted as "not supported" and '1' as "supported".

RAWParamKind:	**Interpret as data type**
RAWParamWhiteBalanceSetting	RemoteFormatWBType 45
RAWParamWhiteBalanceKelvin	-

One example. If `OpenRAWObject_GetRemoteParamSupported(RAWParamWhiteBalanceSetting)` returns '01110000000000000' it shall be interpreted as

0 - RemoteFormatWBNotUsed
1 - RemoteFormatWBAuto
1 - RemoteFormatWBDaylight
1 - RemoteFormatWBCloudy
0 - RemoteFormatWBTungsten
0 - RemoteFormatWBFluorscent
0 - RemoteFormatWBFlash
0 - RemoteFormatWBFluorescentLight
0 - RemoteFormatWBCustom
0 - RemoteFormatWBCustom1
0 - RemoteFormatWBCustom2
0 - RemoteFormatWBBW
0 - RemoteFormatWBShade
0 - RemoteFormatWBKelvin
0 - RemoteFormatWBPCSet1
0 - RemoteFormatWBPCSet2
0 - RemoteFormatWBPCSet3

The conclusion is that Auto, Daylight and Cloudy RAW development whitebalance parameter are supported.

Parameter:

- RAWParamKind: The RAW development parameter to get support data for.

See also:
OpenRAWObject 15

If any errors occurs an ECamException 2 exception will be raised.

3.4.29 TCamRemote.RemoteActivateViewfinderAuto

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

Syntax:
```
procedure RemoteActivateViewfinderAuto;
```

Prerequisite:
The connected camera(s) have been enumrated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
The remote viewfinder must be active, set by the RemoteStartViewfinder 35 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 2 exception will be raised.

3.4.30 TCamRemote.RemoteAFLock

Sets or unsets the AF lock during remote operations

Syntax:
```
procedure RemoteAFLock(AFLock : boolean);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
The camera AF lock can be set or unset. When set the camera will set the AF and thereafter lock the AF, until it is unset. Set AF Lock to reduce the delay between the request to take a picture (using the RemoteTakePicture 36 method) and the time when the picture is taken by the camera.

Please note the following:
- Check the RemoteStart_DoSupportAfLockUnlock 32 for camera support of AFLock.
- AF Lock does not work in camera auto mode. Use program, av, tv or manual mode to enable AF Lock.
- The following camera does **not** support AFLock. PowerShot S100, S300, S110, S30, S40, G1, G2, Pro90 IS, A10, A20, IXY DIGITAL, IXY DIGITAL 200, IXY DIGITAL 300, DIGITAL IXUS, DIGITAL IXUS v, DIGITAL IXUS 300. Most of the EOS cameras does **not** support AFLock.

Parameter:
- AFLock: The AF lock. true = set, false = unset

If any errors occurs an ECamException 2 exception will be raised.

3.4.31 TCamRemote.RemoteEnd

Ends remote mode.

Syntax:
```
procedure RemoteEnd;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must successfully have been set into remote mode using the RemoteStart 30 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 4 method) will still remain. The OnRemoteEvent 40 event will not be called anymore.

If any errors occurs an ECamException 2 exception will be raised.

3.4.32 TCamRemote.RemoteGetNumberOfAvailableShots

Gets the number of expected pictures that can be stored on the camera.

Syntax:
```
function  RemoteGetNumberOfAvailableShots : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
The method returns the calculated number of pictures that can be stored in camera memory card.

Please note the following:

- Remote type 2 protocol standard cameras 5 return the number of shots left on the camera based on the available disk capacity of the host computer they are connected to.

If any errors occurs an ECamException 2 exception will be raised.

3.4.33 TCamRemote.RemoteGetPicture

Gets a remotely taken picture.

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

type  TypeOfPictureType = (TypeOfPictureNo,
                           TypeOfPictureThumbnail,
                           TypeOfPicturePicture);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
A picture has remotely been taken, using a call to the RemoteTakePicture 36 method.

Description for PowerShot:
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 30 method), the thumbnail is received only when calling RemoteGetPicture. The full picture is then received when calling the RemoteGetPicture a second time. The PictureType parameter

is not used.
The OnRemoteGetPictureEvent 42 is used during transfer.

Description for EOS:
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 30 method), the pictures can be received separately calling RemoteGetPicture using the PictureType parameter to define which to get.
The OnRemoteGetPictureEvent 42 is used during transfer.

Please note the following:

- It is not possible to receive any picture data if the ReleaseDataKind parameter in the RemoteStart 30 method is set to ReleaseModeOnlyToCamera.

Parameter:
- FileName: The filename for the picture to receive.
- PictureType: Sets whether the thumbnail of the picture or the actual picture is transferred. Only used by EOS cameras.

If any errors occurs an ECamException 2 exception will be raised.

3.4.34 TCamRemote.RemoteGetRemoteParams

Gets the current used remote parameters from the camera.

Syntax:
```
function RemoteGetRemoteParams(RemoteParamKind : TRemoteParamType 50 ) : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
This method gets the value for the remote parameter set in the RemoteParamKind parameter. The returned value shall be interpreted as follows:

RemoteParamKind:	**Interpret as datatype**
RemoteParamCompQuality	RemoteFormatQualityType 45
RemoteParamCompQualityPic2RAW	RemoteFormatQualityType 45
RemoteParamImageSize	RemoteFormatSizeType 45
RemoteParamImageSizePic2RAW	RemoteFormatSizeType 45
RemoteParamStrobeSetting	RemoteFormatFlashType 45
RemoteParamStrobeCompSetting	RemoteFormatFlashCompType 45
RemoteParamDriveMode	RemoteFormatDriveModeType 45
RemoteParamImageMode	RemoteFormatShootingModeType 45
RemoteParamMLWeiMode	RemoteFormatMLWeiType 45
RemoteParamAFMode	RemoteFormatAFModeType 45
RemoteParamAFDistance	RemoteFormatAFDistType 45
RemoteParamAFFocusingPoint	RemoteFormatAFFocusingPointType 45
RemoteParamAFAssistLight	RemoteFormatAFLightType 45
RemoteParamWhiteBalanceSetting	RemoteFormatWBType 45
RemoteParamWhiteBalanceKelvin	Kelvin degrees
RemoteParamWhiteBalanceShift	See the RemoteGetRemoteParams_WhitebalanceShift 21 method
RemoteParamPictureStyle	See the RemoteGetRemoteParams_PictureStyle 21 method
RemoteParamContrast	RemoteFormatLevelType 45
RemoteParamColorGain	RemoteFormatLevelType 45
RemoteParamSharpness	RemoteFormatLevelType 45
RemoteParamColorSpace	RemoteFormatColorSpaceType 45
RemoteParamISO	RemoteFormatISOType 45
RemoteParamAv	RemoteFormatAVType 45
RemoteParamTv	RemoteFormatTVType 45
RemoteParamExposureCompensation	RemoteFormatExposureCompType 45
RemoteParamPhotoEffect	RemoteFormatPhotoEffectType 45
RemoteParamBeep	RemoteFormatBeepType 45

Parameter:
- RemoteParamKind: The remote parameter to get value for.

If any errors occurs an ECamException 2 exception will be raised.

3.4.34.1 RemoteGetRemoteParams example

3.4.35 TCamRemote.RemoteGetRemoteParams_PictureStyle

Gets the current used picture style remote parameters from the camera.

Syntax:

```
function  RemoteGetRemoteParams_PictureStyle(RemoteParamPictureStyleKind :
TRemoteParamPictureStyleType 50 ) : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
This method gets the value for the picture style remote parameter set in the RemoteParamPictureStyleKind parameter. The returned value shall be interpreted as follows:

RemoteParamKind:	**Interpret as data type**
RemoteParamPictureStyleEnabled	1 = enabled, 0 = disabled
RemoteParamPictureStyleContrast	integer, min = -4, max = 4
RemoteParamPictureStyleSharpness	integer, min = 0, max = 7
RemoteParamPictureStyleSaturation	integer, min = -4, max = 4
RemoteParamPictureStyleSaturationUsed	1 = used, 0 = not used
RemoteParamPictureStyleColorTone	integer, min = -4, max = 4
RemoteParamPictureStyleColorToneUsed	1 = used, 0 = not used
RemoteParamPictureStyleMonochromeFilter	MonochromeFilterType 45
RemoteParamPictureStyleMonochromeFilterUsed	1 = used, 0 = not used
RemoteParamPictureStyleMonochromeTone	MonochromeToneType 45
RemoteParamPictureStyleMonochromeToneUsed	1 = used, 0 = not used

Either use "Saturation + ColorTone" or "MonochromeFilter + MonochromeTone"
for black and white picture styles. Please set the "Prop"Used flags correct, e.g.
SaturationUsed, ColorToneUsed := true
MonochromeFilterUsed, MonochromeToneUsed := false

Parameter:
- RemoteParamPictureStyleKind: The picture style remote parameter to get value for.

If any errors occurs an ECamException 2 exception will be raised.

3.4.36 TCamRemote.RemoteGetRemoteParams_WhitebalanceShift

Gets the current used whitebalance shift remote parameters from the camera.

Syntax:

```
function  RemoteGetRemoteParams_WhitebalanceShift(RemoteParamWhitebalanceShiftKind :
TRemoteParamWhitebalanceShiftType 51 ) : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
This method gets the value for the whitebalance shift parameter set in the RemoteParamWhitebalanceShiftKind parameter. The returned value shall be interpreted as follows:

RemoteParamKind:	Interpret as data type
RemoteParamWhitebalanceShiftWBShiftEnabled	1 = enabled, 0 = disabled
RemoteParamWhitebalanceShiftAmberBlue	integer, min = -9, max = 9. -9 = only blue, 9 = only amber
RemoteParamWhitebalanceShiftGreenMagenta	integer, min = -9, max = 9. -9 = only magenta, 9 = only green

Parameter:
- RemoteParamWhitebalanceShiftKind: The whitebalance shift remote parameter to get value for.

If any errors occurs an ECamException 2 exception will be raised.

3.4.37 TCamRemote.RemoteGetZoomPos_CurrentZoomPos

Gets camera zoom position.

Syntax:

```
function  RemoteGetZoomPos_CurrentZoomPos : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportZoom 31, before calling this method.

Description:
-

Please note the following:

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

See also:
RemoteGetZoomPos_MaxOpticalZoomPos 22, RemoteGetZoomPos_MaxZoomPos 23

If any errors occurs an ECamException 2 exception will be raised.

3.4.38 TCamRemote.RemoteGetZoomPos_MaxOpticalZoomPos

Gets camera maximum optical zoom position.

Syntax:

```
function  RemoteGetZoomPos_MaxOpticalZoomPos : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportZoom 31, before calling this method.

Description:
-

Please note the following:

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

See also:
RemoteGetZoomPos_CurrentZoomPos 22, RemoteGetZoomPos_MaxZoomPos 23

If any errors occurs an ECamException 2 exception will be raised.

3.4.39 TCamRemote.RemoteGetZoomPos_MaxZoomPos

Gets camera maximum zoom position (including digital magnification).

Syntax:

```
function  RemoteGetZoomPos_MaxZoomPos  : integer;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportZoom 31, before calling this method.

Description:
-

Please note the following:

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

See also:
RemoteGetZoomPos_CurrentZoomPos 22, RemoteGetZoomPos_MaxOpticalZoomPos 22

If any errors occurs an ECamException 2 exception will be raised.

3.4.40 TCamRemote.RemoteLoadCameraRemoteParams

Loads remote parameters from a file and sets these remote parameters in the camera.

Syntax:

```
procedure RemoteLoadCameraRemoteParams(FileName : string);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

Parameter:
- FileName: The filename of the file used to save/load remote parameters.

If any errors occurs an ECamException 2 exception will be raised.

3.4.41 TCamRemote.RemoteSaveCameraRemoteParams

Reads remote parameters from the camera and save these to a file.

Syntax:

```
procedure RemoteLoadCameraRemoteParams(FileName : string);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

Parameter:
- FileName: The filename of the file used to save/load remote parameters.

If any errors occurs an ECamException 2 exception will be raised.

3.4.42 TCamRemote.RemoteSetRemoteParams

Sets remote parameters to the camera.

Syntax:

```
procedure RemoteSetRemoteParams(RemoteParamKind : TRemoteParamType 50 ;
                                Value           : integer);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31 , before calling this method.

Description:
Supported remote parameters from the camera can be received using the RemoteStart_GetRemoteParamSupported 33 method. These parameters have been verified for camera acceptance 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 19 method to verify which parameters that are accepted or not.

It is only possible to set one remote parameter at a time. Use the RemoteParamKind to set which parameter to set and value parameter as the ordinal of the enums listed below:

RemoteParamKind:	**Interpret as datatype**
RemoteParamCompQuality	See the RemoteSetRemoteParams_ImageQuality 27 method
RemoteParamCompQualityPic2RAW	See the RemoteSetRemoteParams_ImageQuality 27 method
RemoteParamImageSize	See the RemoteSetRemoteParams_ImageQuality 27 method
RemoteParamImageSizePic2RAW	See the RemoteSetRemoteParams_ImageQuality 27 method
RemoteParamStrobeSetting	RemoteFormatFlashType 45
RemoteParamStrobeCompSetting	RemoteFormatFlashCompType 45
RemoteParamDriveMode	RemoteFormatDriveModeType 45
RemoteParamImageMode	RemoteFormatShootingModeType 45
RemoteParamMLWeiMode	RemoteFormatMLWeiType 45
RemoteParamAFMode	RemoteFormatAFModeType 45
RemoteParamAFDistance	RemoteFormatAFDistType 45
RemoteParamAFFocusingPoint	RemoteFormatAFFocusingPointType 45
RemoteParamAFAssistLight	RemoteFormatAFLightType 45
RemoteParamWhiteBalanceSetting	RemoteFormatWBType 45
RemoteParamWhiteBalanceKelvin	Kelvin degrees
RemoteParamWhiteBalanceShift	See the RemoteSetRemoteParams_WhitebalanceShift 28 method
RemoteParamPictureStyle	See the RemoteSetRemoteParams_PictureStyle 27 method
RemoteParamContrast	RemoteFormatLevelType 45
RemoteParamColorGain	RemoteFormatLevelType 45
RemoteParamSharpness	RemoteFormatLevelType 45
RemoteParamColorSpace	RemoteFormatColorSpaceType 45
RemoteParamISO	RemoteFormatISOType 45
RemoteParamAv	RemoteFormatAVType 45
RemoteParamTv	RemoteFormatTVType 45
RemoteParamExposureCompensation	RemoteFormatExposureCompType 45
RemoteParamPhotoEffect	RemoteFormatPhotoEffectType 45
RemoteParamBeep	RemoteFormatBeepType 45

Please note the following:

- It is not recommended that the viewfinder is on when setting remote parameters using the RemoteSetRemoteParams method. The parameters will be set but it will take time, since the viewfinder may be restarted for each new parameter set. Therefore turn off the viewfinder before setting remote parameters, then turn it on again after the parameters have been set.
- Newer PowerShot cameras do have problems handling not supported remote parameters. The camera may shut down, if not supported parameters are used.

Parameter:
- **RemoteParamKind:** The remote parameter to the camera.
- **Value:** The remote parameter (ordinal) value.

See also:
RemoteSetRemoteParams_ImageQuality 27, RemoteSetRemoteParams_PictureStyle 27, RemoteSetRemoteParams_WhitebalanceShift 28

If any errors occurs an ECamException 2 exception will be raised.

3.4.42.1 RemoteSetRemoteParams example

3.4.43 TCamRemote.RemoteSetRemoteParams_ImageQuality

Sets image quality remote parameters to the camera.

Syntax:
```
procedure RemoteSetRemoteParams_ImageQuality(ImageSize : integer;
                                             ImageQuality : integer;
                                             ImageSizePic2RAW : integer;
                                             ImageQualityPic2RAW : integer);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
-

Please note the following:
See same section in RemoteSetRemoteParams 24

Parameter:
- ImageSize: The image size of the main picture. Interpret as the ordinal value of RemoteFormatSizeType 45.
- ImageQuality: The image quality of the main picture. Set to RemoteFormatQualityRAW if RAW or RAW+JPEG is wanted. Interpret as the ordinal value of RemoteFormatQualityType 45.
- ImageSizePic2RAW: The image size of the JPEG picture when taking a RAW+JPEG picture. Interpret as the ordinal value of RemoteFormatSizeType 45.
- ImageQualityPic2RAW: The image quality of the JPEG picture when taking a RAW+JPEG picture. Interpret as the ordinal value of RemoteFormatQualityType 45.

See also:
RemoteSetRemoteParams 24, RemoteSetRemoteParams_PictureStyle 27, RemoteSetRemoteParams_WhitebalanceShift 28

If any errors occurs an ECamException 2 exception will be raised.

3.4.44 TCamRemote.RemoteSetRemoteParams_PictureStyle

Sets picture style remote parameter to the camera.

Syntax:
```
procedure RemoteSetRemoteParams_PictureStyle(PictureStyleEnabled  : boolean;
                                             Contrast             : IntN4_4Type 45;
                                             Sharpness            : Int0_7Type 45;
                                             Saturation           : IntN4_4Type 45;
                                             SaturationUsed       : boolean;
                                             ColorTone            : IntN4_4Type 45;
                                             ColorToneUsed        : boolean;
                                             MonochromeFilter     : MonochromeFilterType 45;
                                             MonochromeFilterUsed : boolean;
                                             MonochromeTone       : MonochromeToneType 45;
                                             MonochromeToneUsed   : boolean);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
Sets picture style remote parameters to the camera.

Either use "Saturation + ColorTone" or "MonochromeFilter + MonochromeTone" for black and white picture styles. Please set the "Prop"Used flags correct, e.g.
SaturationUsed, ColorToneUsed := true
MonochromeFilterUsed, MonochromeToneUsed := false

Please note the following:
See same section in RemoteSetRemoteParams 24

Parameter:
- See Syntax:

See also:
RemoteSetRemoteParams 24, RemoteSetRemoteParams_ImageQuality 27, RemoteSetRemoteParams_WhitebalanceShift 28

If any errors occurs an ECamException 2 exception will be raised.

3.4.45 TCamRemote.RemoteSetRemoteParams_WhitebalanceShift

Sets whitebalance shift remote parameter to the camera.

Syntax:
```
procedure RemoteSetRemoteParams_WhitebalanceShift(WBShiftEnabled : boolean;
                                                   AmberBlue      : IntN9_9Type;
                                                   GreenMagenta   : IntN9_9Type);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check RemoteStart_DoSupportShootingPara 31, before calling this method.

Description:
-

Please note the following:
See same section in RemoteSetRemoteParams 24

Parameter:
- **WBShiftEnabled:** Set to true enables whitebalance shift remote parameter.
- **AmberBlue:** -9 = only blue, 9 = only amber
- **GreenMagenta:** -9 = only magenta, 9 = only green

See also:
RemoteSetRemoteParams 24, RemoteSetRemoteParams_ImageQuality 27, RemoteSetRemoteParams_PictureStyle 27

If any errors occurs an ECamException 2 exception will be raised.

3.4.46 TCamRemote.RemoteSetViewfinderOutput

Changes the remote viewfinder output destination.

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

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

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

The camera must be in remote mode, set by the RemoteStart 30 method.
The remote viewfinder must be active, set by the RemoteStartViewfinder 35.

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 2 exception will be raised.

3.4.47 TCamRemote.RemoteSetZoomPos

Sets camera zoom position.

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

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check DoSupportZoom in RemoteFuncType returned by the Connect 4 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 RemoteGetZoomPos_MaxZoomPos 23

If any errors occurs an ECamException 2 exception will be raised.

3.4.47.1 RemoteSetZoomPos example

3.4.48 TCamRemote.RemoteStart

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

Syntax:
```
procedure RemoteStart(ReleaseMode           : ReleaseModeType;
                      ReleaseDataKind       : ReleaseDataKindType;
                      ProbeRemoteParameters : boolean;
                      CacheRemoteParamDir   : string);

type ReleaseModeType =  (ReleaseModeOnlyToPC,
                         ReleaseModeBothPCAndCamera,
                         ReleaseModeOnlyToCamera);

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

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
Call the RemoteSupported 15 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.The camera remote capability parameters including supported remote parameters can be received using the RemoteStart_XXX methods. The OnRemoteEvent 40 event will be called, when a remote camera event has occurred.
The time needed for TCamRemote to probe the camera, at each startup, may be shortened if a temporary cache dir is set to the CacheRemoteParamDir parameter. When the cache function is used, TCamRemote will first look into the CacheRemoteParamDir directory and see if there are any stored supported parameters for the connected camera (camera mode is also considered). If found the supported remote parameters are read from the file and is immediately assigned the `RemoteParamSupported` in the record returned from this method, and no further probing is needed. If no cache file for the connected camera is found, the camera is probed and the supported parameters found are stored in a cache file for later use.

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 are not supported by the EOS cameras. These parameters must be set manually on the camera before pictures are taken.
- PowerShot cameras do always default use 'Program' camera mode when connected, even if the knobs on the PowerShot camera is set to something else. It is possible to set the camera mode (or image mode) to other values (e.g. manual, TV or AV) using the RemoteSetRemoteParams 24 method. EOS cameras must set the camera mode using the camera knobs before starting remote handling of the camera.
- Newer PowerShot cameras do have problems handling not supported remote parameters. The camera may shut down, if not supported parameters are used.

Parameter:
- ReleaseMode: Sets pictures destination (on the PC and/or camera) used when calling RemoteTakePictures 36.
- ReleaseDataKind: Sets if thumbnail and/or picture shall be taken when calling RemoteTakePicture 36.
- ProbeRemoteParameters: Sets if remote parameters is to be probed or not. Remote mode will start more quickly if remote parameters not are probed, however it is recommended to probe remote parameters.
- CacheRemoteParamDir: Sets the directory where cached remote parameters are stored. If set

to '' the cache parameters function is not used.

See also:
RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportShootingPara 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_ReqViewfinderOffWhenShooting 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.48.1 RemoteStart example

```
  //Check if remote mode is supported
  if (not CamRemoteActiveX.RemoteSupported) then
  begin
    ECamException.Create('The connected camera does not support remote mode');
  end;
  StatusBar.SimpleText := 'Starts remote process and probes remote parameters';
  //Starts the remote mode
  Log('Opens remote mode');
  CamRemoteActiveX.RemoteStart(Ord(mReleaseMode),
                               Ord(REMOTE_MODE),
                               CheckBoxProbeRemoteParameters.Checked,
                               GetEnvironmentVariable('Temp') + '\' + REMOTE_SUBKEY);
  mCameraCapability.DoSupportZoom                :=
CamRemoteActiveX.RemoteStart_DoSupportZoom;
  mCameraCapability.DoSupportShootingPara        :=
CamRemoteActiveX.RemoteStart_DoSupportShootingPara;
  mCameraCapability.DoSupportViewfinder          :=
CamRemoteActiveX.RemoteStart_DoSupportViewfinder;
  mCameraCapability.ReqViewfinderOffWhenShooting :=
CamRemoteActiveX.RemoteStart_ReqViewfinderOffWhenShooting;
  mCameraCapability.DoSupportAfLockUnlock        :=
CamRemoteActiveX.RemoteStart_DoSupportAfLockUnlock;
```

3.4.49 TCamRemote.RemoteStart_DoSupportZoom

Returns if the connected camera support remote zooming.

Syntax:
`function  RemoteStart_DoSupportZoom : boolean;`

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

See also:
RemoteStart 30, RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportShootingPara 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_ReqViewfinderOffWhenShooting 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.50 TCamRemote.RemoteStart_DoSupportShootingPara

Returns if the connected camera support setting remote parameters.

Syntax:
`function  RemoteStart_DoSupportShootingPara : boolean;`

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

See also:
RemoteStart 30, RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_ReqViewfinderOffWhenShooting 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.51 TCamRemote.RemoteStart_DoSupportViewfinder

Returns if the connected camera support remote handling of the viewfinder.

Syntax:
```
function  RemoteStart_DoSupportViewfinder : boolean;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

See also:
RemoteStart 30, RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportShootingPara 31, RemoteStart_ReqViewfinderOffWhenShooting 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.52 TCamRemote.RemoteStart_ReqViewfinderOffWhenShooting

Returns if the connected camera remote viewfinder must be disabled before taking a picture remotely.

Syntax:
```
function  RemoteStart_ReqViewfinderOffWhenShooting : boolean;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

See also:
RemoteStart 30, RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportShootingPara 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.53 TCamRemote.RemoteStart_DoSupportAfLockUnlock

Returns if the connected camera supports the AF-lock remote parameter function.

Syntax:
```
function  RemoteStart_DoSupportAfLockUnlock : boolean;
```

Prerequisite:

The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
-

See also:
RemoteStart 30, RemoteStart_GetRemoteParamSupported 33, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportShootingPara 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_ReqViewfinderOffWhenShooting 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.54 TCamRemote.RemoteStart_GetRemoteParamSupported

Gets the supported remote parameters that are supported by the connected camera.

Syntax:

```
function  RemoteStart_GetRemoteParamSupported(RemoteParamKind : TRemoteParamType 50) : string;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description:
This method gets the supported remote parameters that are supported by the connected camera. These supported remote parameters are probed in the RemoteStart 30 method, if the ProbeRemoteParameters is set to true. One parameters at a time can be received, set by the RemoteParamKind parameter. The returned value is a string which length is equal the data type listed below (e.g. RemoteFormatWBType 45 for RemoteParamWhiteBalanceSetting). Each character in the string is either a '0' or '1'. '0' is interpreted as "not supported" and '1' as "supported".

RemoteParamKind:	**Interpret as datatype**
RemoteParamCompQuality	RemoteFormatQualityType 45
RemoteParamCompQualityPic2RAW	RemoteFormatQualityType 45
RemoteParamImageSize	RemoteFormatSizeType 45
RemoteParamImageSizePic2RAW	RemoteFormatSizeType 45
RemoteParamStrobeSetting	RemoteFormatFlashType 45
RemoteParamStrobeCompSetting	RemoteFormatFlashCompType 45
RemoteParamDriveMode	RemoteFormatDriveModeType 45
RemoteParamImageMode	RemoteFormatShootingModeType 45
RemoteParamMLWeiMode	RemoteFormatMLWeiType 45
RemoteParamAFMode	RemoteFormatAFModeType 45
RemoteParamAFDistance	RemoteFormatAFDistType 45
RemoteParamAFFocusingPoint	RemoteFormatAFFocusingPointType 45
RemoteParamAFAssistLight	RemoteFormatAFLightType 45
RemoteParamWhiteBalanceSetting	RemoteFormatWBType 45
RemoteParamWhiteBalanceKelvin	-
RemoteParamWhiteBalanceShift	Is either '0' (not supported) or '1' (supported)
RemoteParamPictureStyle	Is either '0' (not supported) or '1' (supported)
RemoteParamContrast	RemoteFormatLevelType 45
RemoteParamColorGain	RemoteFormatLevelType 45
RemoteParamSharpness	RemoteFormatLevelType 45
RemoteParamColorSpace	RemoteFormatColorSpaceType 45
RemoteParamISO	RemoteFormatISOType 45
RemoteParamAv	RemoteFormatAVType 45
RemoteParamTv	RemoteFormatTVType 45
RemoteParamExposureCompensation	RemoteFormatExposureCompType 45
RemoteParamPhotoEffect	RemoteFormatPhotoEffectType 45
RemoteParamBeep	RemoteFormatBeepType 45

One example. If `RemoteStart_GetRemoteParamSupported(RemoteParamWhiteBalanceSetting)` returns '0111000000000000' it shall be interpreted as

0 - RemoteFormatWBNotUsed
1 - RemoteFormatWBAuto
1 - RemoteFormatWBDaylight
1 - RemoteFormatWBCloudy
0 - RemoteFormatWBTungsten
0 - RemoteFormatWBFluorscent
0 - RemoteFormatWBFlash
0 - RemoteFormatWBFluorescentLight
0 - RemoteFormatWBCustom
0 - RemoteFormatWBCustom1
0 - RemoteFormatWBCustom2
0 - RemoteFormatWBBW
0 - RemoteFormatWBShade
0 - RemoteFormatWBKelvin
0 - RemoteFormatWBPCSet1
0 - RemoteFormatWBPCSet2

0 - RemoteFormatWBPCSet3

The conclusion is that Auto, Daylight and Cloudy whitebalance remote parameter are supported.

Please note the following:

- Do not call this method if the camera not supports operation of remote parameters. Use the RemoteStart_DoSupportShootingPara 31 method to check the camera for support.

Parameter:
- RemoteParamKind: The remote parameter to get support data for.

See also:
RemoteStart 30, RemoteStart_DoSupportZoom 31, RemoteStart_DoSupportShootingPara 31, RemoteStart_DoSupportViewfinder 32, RemoteStart_ReqViewfinderOffWhenShooting 32, RemoteStart_DoSupportAfLockUnlock 32

If any errors occurs an ECamException 2 exception will be raised.

3.4.55 TCamRemote.RemoteStartViewfinder

Starts the remote viewfinder.

Syntax:
```
procedure RemoteStartViewfinder;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
Check DoSupportViewfinder in RemoteFuncType returned by the Connect 4 method, before calling this method.

Description:
The camera starts the electronic remote viewfinder. The camera will take 320x240 pictures rapidly and call the OnViewfinderEvent 43 when a picture is available.

Please note the following:

- It is not recommended that the viewfinder is on when setting remote parameters using the RemoteSetRemoteParams 24 method. The parameters will be set but it will take time, since the viewfinder may be restarted for each new parameter set. Therefore turn off the viewfinder before setting remote parameters, then turn it on again after the parameters have been set.
- EOS cameras does not support remote viewfinder.

If any errors occurs an ECamException 2 exception will be raised.

3.4.55.1 RemoteStartViewfinder example

```
//Check if remote viewfinder is supported
if (not CamRemote.RemoteStart_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;
```

3.4.56 TCamRemote.RemoteStopViewfinder

Stops the remote viewfinder.

Syntax:
```
procedure RemoteStopViewfinder;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
The camera remote viewfinder must be active, after calling the RemoteStartViewfinder 35 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 2 exception will be raised.

3.4.57 TCamRemote.RemoteSupported

Returns true if the camera supports remote mode.

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

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

Description:
Use this function before starting the remote mode, by calling the RemoteStart 30 method.

If any errors occurs an ECamException 2 exception will be raised.

3.4.58 TCamRemote.RemoteTakePicture

Takes a picture and returns the number of pictures that is ready to be read, using the RemoteGetPicture 18 method.

Syntax:
```
procedure RemoteTakePicture;
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 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 30 method. RemoteTakePicture will return immediately when the request has been sent to the camera.

Please note the following:
- It is possible to take a picture when using remote viewfinder, only if the RemoteStart_ReqViewfinderOffWhenShooting 32 method returns false.
- The RemoteTakePicture method will return as soon as a request to the camera to take the picture is sent. The OnRemoteEvent 40 event will be called twice when taking a picture. First with RemoteEventCallbackReleaseStart 44 (with exception for newer EOS cameras including the EOS 20D) when the picture is taken and RemoteEventCallbackReleaseComplete 44 (for older EOS cameras and PowerShot cameras) or RemoteEventCallbackReleaseImageReady 44 (only for newer EOS cameras including the EOS 20D) when the picture data is available to be read.

If any errors occurs an ECamException 2 exception will be raised.

3.4.58.1 RemoteTakePicture example

```
procedure TFormRemote.TakeThePicture;
begin
  //Starts remote mode
  mCameraCapability := CamRemote.RemoteStart(ReleaseModeOnlyToPC,
                                             ReleaseDataKingTakeBothThumbAndPic,
                                             true);

  //Take a picture.
  CamRemote.RemoteTakePicture;
end;
```

3.4.59 TCamRemote.SetOwnerName

Sets the owner name in the camera.

Syntax:

```
procedure SetOwnerName(OwnerName : String);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method.
A successful connection must have been established with the Connect 4 method.

Description:
Sets the owner name in the camera.

Parameter:
- **OwnerName:** The owner name to be set.

If any errors occurs an ECamException 2 exception will be raised.

3.4.60 TCamRemote.SetRAWDevelopmentParameters

Sets RAW development parameters for the RAW object.

Syntax:

```
procedure SetRAWDevelopmentParameters(RAWParamKind : TRAWParamType;
                                      Value : integer);
```

Prerequisite:
The RAW object has successfully been created using the OpenRAWObject 15 method.

Description:
It is only possible to set one RAW development parameter at a time. Use the RAWParamKind to set which parameter to set and value parameter as the ordinal of the enums listed below:

RAWParamKind:	Interpret as datatype
RAWParamWhiteBalanceSetting	RemoteFormatWBType 45
RAWParamWhiteBalanceKelvin	Kelvin degress. Only valid if the RAWParamKind parameter is set to RemoteFormatWBKelvin in the camera.degrees

Parameter:
- **RAWParamKind:** The RAW development parameter to set.
- **Value:** The RAW development parameter (ordinal) value..

If any errors occurs an ECamException 2 exception will be raised.

3.4.61 TCamRemote.SetTimeInCamera

Sets time in the camera.

Syntax:

```
procedure SetTimeInCamera(YearCam        : integer;
                          MonthCam       : integer;
                          DayCam         : integer;
                          HourCam        : integer;
```

```
                                  MinutesCam     : integer;
                                  SecondCam      : integer;
                                  MillisecondCam : integer);
```

Prerequisite:
The connected camera(s) have been enumerated using the OpenCameraEnumeration 14 method. A successful connection must have been established with the Connect 4 method.

Description:
Sets the camera time to the time in the time parameters.

Parameter:
- The parameters are quite easy to understand and defines the time to set.

If any errors occurs an ECamException 2 exception will be raised.

3.5 TCamRemote events

3.5.1 List of events in TCamRemote

OnEvent 38
OnGetPictureEvent 39
OnRawDevelopEvent 39
OnRemoteEvent 40
OnRemoteGetPictureEvent 42
OnRemoteProbeParamEvent 42
OnRemoteTakePictureEvent 43
OnViewfinderEvent 43

3.5.2 TCamRemote.OnEvent

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

Syntax:

```
property OnEvent:TNotifyOCXCameraEvent

TNotifyOCXCameraEvent=procedure(Severity : EventSeverityType 44 ;
                                Event    : EventEnumType 44 ) of object;
```

Prerequisite:
A successful connection must have been established with the Connect 4 method.

Description:
-

3.5.2.1 OnEvent example

```
//Camera event handler
procedure TFormRemote.CamRemoteActiveXEvent(ASender: TObject; Severity,
  Event: TOleEnum);
var camera_text   : string;
    text_message  : string;
    message_out   : MessageQueueElementType;
begin
  FormRemote.Log('Callback Severity=' + inttostr(Ord(Severity)) +
                 'Event =' + inttostr(Ord(Event)));
  //Do not handle information message, not of interest
  if (Severity = Ord(EventSeverityInfo)) then
    exit;
  camera_text := 'Camera event-';
  case Severity of
    Ord(EventSeverityWarning) : text_message := 'Camera warning event';
    Ord(EventSeverityClosing) : text_message := 'Camera shutdown event';
  end;
  case Event of
    Ord(EventBatteryLevelNormal)           : camera_text := camera_text + 'Battery level
normal';
```

```
    Ord(EventBatteryLevelWeak)             : camera_text := camera_text + 'Battery level
weak';
    Ord(EventBatteryLevelSafetyLow)        : camera_text := camera_text + 'Battery level
safety low';
    Ord(EventBatteryLevelLB)               : camera_text := camera_text + 'Battery level LB';
    Ord(EventDialChanged)                  : camera_text := camera_text + 'Dial changed';
    Ord(EventCFGateOpened)                 : camera_text := camera_text + 'CF gate opened';
    Ord(EventBatteryCoverOpened)           : camera_text := camera_text + 'Battery cover
opened';
    Ord(EventConnectionDisappeared)        : camera_text := camera_text + 'Camera connection
disappeared';
    Ord(EventUnrecoverableError)           : camera_text := camera_text + 'Unrecoverable
error';
    Ord(EventUnkonwnCommandReceived)       : camera_text := camera_text + 'Unknown command
received';
    Ord(EventRemoteParameterChanged)       : camera_text := camera_text + 'Remote parameter
changed';
    Ord(EventRemoteCaptureError)           : camera_text := camera_text + 'Capture error';
    Ord(EventRemoteShutdownReasonNotKnown) : camera_text := camera_text + 'Shutdown reason not
known';
  end;
  //Log the camera event
  FormRemote.Log(camera_text);
end;
```

3.5.3 TCamRemote.OnGetPictureEvent

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

Syntax:

```
property OnGetPictureEvent: TNotifyGetPictureEvent;

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

Prerequisite:
A successful connection must have been established with the Connect 4 method.
A successful call to the OpenCameraCollection 12 method, to list pictures on the camera.
The picture is received using the GetPicture 10 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 10 method.

3.5.3.1 OnGetPictureEvent example

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

3.5.4 TCamRemote.OnRawDevelopEvent

Occurs when the TIFF or JPEG file is developed from the RAW object.

Syntax:

```
property OnRawDevelopEvent:TNotifyRawDevelopEvent

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

Prerequisite:
The RAW object has successfully been created using the OpenRAWObject 15 method.
The TIFF/JPEG file is developed from the RAW object using the DevelopRAWPicture 9 method.

Description:
This event is a callback of progress during TIFF/JPEG file creation from the RAW object when using

the DevelopRAWPicture 9 method.

3.5.4.1 OnRawDevelopEvent example

```
procedure TFormMain.CamRemoteRawDevelopEvent(PercentageDone: Integer);
begin
   ProgressBar.Position := PercentageDone;
end;
```

3.5.5 TCamRemote.OnRemoteEvent

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

Syntax:

```
property OnRemoteEvent: TNotifyOCXRemoteEvent;

TNotifyOCXRemoteEvent=procedure(Event         : RemoteEventCallbackType;
                                EventValid    : NotifyRemoteEventValidType; //Not currently
used
                                NumOfEvents   : integer;
                                TypeOfPicture : TypeOfPictureType) of object;

type
  RemoteEventCallbackType = (RemoteEventCallbackNotUsed,
                             RemoteEventCallbackReleaseStart,
                             RemoteEventCallbackReleaseComplete,
                             RemoteEventCallbackResetHWError,
                             RemoteEventCallbackChangedByUI,
                             RemoteEventCallbackCamReleaseOn,
                             RemoteEventCallbackViewfinderOn,
                             RemoteEventCallbackViewfinderOff,
                             RemoteEventCallbackReleaseImageReady);
  NotifyRemoteEventValidType = (NoNotifyRemoteEventValid,
                                NotifyRemoteEventValid1,
                                NotifyRemoteEventValid2);
  TypeOfPictureType = (TypeOfPictureNo,
                       TypeOfPictureThumbnail,
                       TypeOfPicturePicture);
```

Prerequisite:
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.

Description of RemoteEventCallbackType:
The interpretation of RemoteEventCallbackType is:

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 18 method. Used by PowerShot cameras and EOS camera older than the 20D. The number of pictures to receive is stored in the NumOfEvents parameter data.
RemoteEventCallbackResetHWError	A hardware error has occurred.
RemoteEventCallbackChangedByUI	The remote parameters has been changed manually in the camera. Reread new remote parameters using the RemoteGetRemoteParams 19 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 36 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.
RemoteEventCallbackReleaseImageReady	Picture or thumbnail taken remotely is available to be read using the RemoteGetPicture 18 method. Used only by EOS camera newer and including the 20D. The TypeOfPicture parameter is used to get type of picture to get.

3.5.5.1 OnRemoteEvent example

```
procedure TFormRemote.CamRemoteActiveXRemoteEvent(ASender: TObject; Event,
  EventValid: TOleEnum; NumOfEvents: Integer; TypeOfPicture: TOleEnum);
begin
  FormRemote.Log('Remote callback event=' + inttostr(Ord(Event)));
  //Check if the release button was pressed on the camera.
  //This applies to PowerShot cameras. EOS cameras do not generate
  //this event when pressing the shutter butten. Instead the EOS camera
  //takes the picture and generates a RemoteEventCallbackReleaseStart event.
  case Event of
    Ord(RemoteEventCallbackCamReleaseOn) :
      begin
        TimerTakePicture.Enabled := true;
      end;
    Ord(RemoteEventCallbackReleaseComplete) :
      begin
        case REMOTE_MODE of
          ReleaseDataKingTakeBothThumbAndPic :
            begin
              mThumbnailsToReceive := mThumbnailsToReceive + NumOfEvents;
              mPicturesToReceive   := mPicturesToReceive   + NumOfEvents;
            end;
          ReleaseDataKindTakeOnlyPicture :
            begin
              mPicturesToReceive   := mPicturesToReceive   + NumOfEvents;
            end;
        end;
      end;
    Ord(RemoteEventCallbackReleaseImageReady) :
      begin
        case TypeOfPicture of
```

```
          Ord(TypeOfPictureThumbnail) : mThumbnailsToReceive := mThumbnailsToReceive + 1;
          Ord(TypeOfPicturePicture)   : mPicturesToReceive   := mPicturesToReceive   + 1;
        end;
      end;
    Ord(RemoteEventCallbackChangedByUI) :
      begin
        if (FormRemote.ReleaseProcessActive) then
        begin
          FormRemote.ButtonGetCamSetClick(Application);
        end;
      end;
  end;
end;
```

3.5.6 TCamRemote.OnRemoteGetPictureEvent

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

Syntax:

```
property OnRemoteGetPictureEvent: TNotifyRemoteGetPictureEvent;

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

Prerequisite:
A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
A picture must have been taken using the RemoteTakePicture 36 method.
The picture is received using the RemoteGetPicture 18 method.

Description:
This event is a callback of progress during file transfer when using the RemoteGetPicture 18 method.

3.5.6.1 OnRemoteGetPictureEvent example

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

3.5.7 TCamRemote.OnRemoteProbeParamEvent

Occurs when probing remote parameters on a camera.

Syntax:

```
property OnRemoteProbeParamEvent: TNotifyRemoteProbeParamEvent;

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

Prerequisite:
A successful connection must have been established with the Connect 4 method.
A successful call to the RemoteStart 30 method, to start camera remote mode and probing the camera for remote parameters.

Description:
This event is a callback of progress during probing of remote parameters in the RemoteStart 30 method.

3.5.7.1 OnRemoteProbeParamEvent example

```
procedure TFormRemote.CamRemoteRemoteProbeParamEvent(
  PercentageDone: Integer);
begin
  FormRemote.StatusBar.SimpleText := 'Probes remote parameters. ' +
                                     IntToStr(PercentageDone) +
                                     ' percent done.';
  FormRemote.ProgressBar.Position := PercentageDone;
end;
```

3.5.8 TCamRemote.OnRemoteTakePictureEvent

Occurs when transferring data from the camera to the PC.

Syntax:

```
property OnRemoteTakePictureEvent: TNotifyRemoteTakePictureEvent;

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

Prerequisite:

A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
A picture is taken using the RemoteTakePicture 36 method.

Description:

This event is a callback of progress during file transfer from the camera to the PC when using the RemoteTakePicture 36 method.

3.5.8.1 OnRemoteTakePictureEvent example

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

3.5.9 TCamRemote.OnViewfinderEvent

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

Syntax:

```
property OnViewfinderEvent: TNotifyOCXViewfinderEvent;

TNotifyOCXViewfinderEvent=procedure(ViewfinderFileName : string) of object;
```

Prerequisite:

A successful connection must have been established with the Connect 4 method.
The camera must be in remote mode, set by the RemoteStart 30 method.
The remote viewfinder must be active, set by the RemoteStartViewfinder 35.

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. The user is responsible to delete the viewfinder file with filename as set by the ViewfinderFileName parameter. The viewfinder file is guaranteed to have a unique name and is never touch by the TCamRemote after calling this event.

3.5.9.1 OnRemoteViewfinder example

```
procedure TFormRemote.CamRemoteActiveXViewfinderEvent(ASender: TObject;
  const ViewfinderFileName: WideString);
var jpg : TJpegImage;
begin
  if (not mNewViewfinder) then
  begin
    jpg := TJpegImage.Create;
     //Get the 320x240 picture and save it into a Bitmap
    try
      jpg.LoadFromFile(ViewfinderFileName);
      mViewfinderPic.Assign(jpg);
    finally
      jpg.Free;
    end;
    mNewViewfinder := true;
  end;
  DeleteFile(ViewfinderFileName);
end;
```

3.6 TCamRemote types

3.6.1 BatterySourceType

Syntax:

```
BatterySourceType = (BatterySourceUnknown,
                     BatterySourceAC,
                     BatterySourceLitium,
                     BatterySourceNiMH,
                     BatterySourceNiCD,
                     BatterySourceAlMN);
```

3.6.2 BatteryStatusType

Syntax:

```
BatteryStatusType = (BatteryStatusUnknown,
                     BatteryStatusNormal,
                     BatteryStatusWeak,
                     BatteryStatusSafetyLow);
```

3.6.3 CamModType

Syntax:

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

Description

Used to define if PowerShot or EOS camera is used.

3.6.4 EventCallbackType

Syntax:

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

EventEnumType     = (EventNotUsed,
                     EventBatteryLevelNormal,
                     EventBatteryLevelWeak,
                     EventBatteryLevelSafetyLow,
                     EventBatteryLevelLB,
                     EventDialChanged,
                     EventCFGateOpened,
                     EventBatteryCoverOpened,
                     EventConnectionDisappeared,
                     EventUnrecoverableError,
                     EventUnkonwnCommandReceived,
                     EventRemoteParameterChanged,
                     EventRemoteCaptureError,
                     EventRemoteShutdownReasonNotKnown,
                     EventRemoteNewSupportedParameters);
```

Description

EventSeverityType and EventEnumType are used as a parameter to the OnEvent 38 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.

3.6.5 RemoteEventCallbackType

Syntax:

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

Description

RemoteEventCallbackType is used as a parameter to the OnRemoteEvent 40 event. The interpretation of RemoteEventCallbackType is:

`RemoteEventCallbackReleaseStart`	A take picture remote request is sent to the camera.
`RemoteEventCallbackReleaseComplete`	Pictures that have been remotely taken can be received using the RemoteGetPicture 18 method.
`RemoteEventCallbackResetHWError`	Hardware error.
`RemoteEventCallbackChangedByUI`	The user has change remote parameters manually in the camera. Use the RemoteGetRemoteParams 19 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 18 method.
`RemoteEventCallbackViewfinderOn`	The remote viewfinder is on.
`RemoteEventCallbackViewfinderOff`	The remote viewfinder is off.
`RemoteEventCallbackReleaseImageReady`	Same as `RemoteEventCallbackReleaseComplete` but only valid for newer EOS cameras ("EOS 20D and forward").

3.6.6 RemoteReleaseParametersType

Syntax:

```
//Remote Format Quality
type RemoteFormatQualityType = (RemoteFormatQualityNotUsed,
                                RemoteFormatQualityEconomy,
                                RemoteFormatQualityNormal,
                                RemoteFormatQualityFine,
                                RemoteFormatQualitySuperfine,
                                RemoteFormatQualityRAW);

//Remote Format Size
type RemoteFormatSizeType = (RemoteFormatSizeNotUsed,
                             RemoteFormatSizeLarge,
                             RemoteFormatSizeMedium,
                             RemoteFormatSizeSmall,
                             RemoteFormatSizeMedium1,
                             RemoteFormatSizeMedium2,
                             RemoteFormatSizeMedium3);

//Remote Shooting mode
type RemoteFormatShootingModeType = (RemoteFormatShootingModeNotUsed,
                                     RemoteFormatShootingModeAuto,
                                     RemoteFormatShootingModeManual,
                                     RemoteFormatShootingModeFarScene,
                                     RemoteFormatShootingModeFastShutter,
                                     RemoteFormatShootingModeSlowShutter,
                                     RemoteFormatShootingModeNightScene,
                                     RemoteFormatShootingModeGrayScene,
                                     RemoteFormatShootingModeSepia,
                                     RemoteFormatShootingModePortrait,
                                     RemoteFormatShootingModeSport,
                                     RemoteFormatShootingModeMacro,
                                     RemoteFormatShootingModeBW,
                                     RemoteFormatShootingModePanFocus,
                                     RemoteFormatShootingModeVivid,
                                     RemoteFormatShootingModeNeutral,
                                     RemoteFormatShootingModeProgram,
                                     RemoteFormatShootingModeTV,
                                     RemoteFormatShootingModeAV,
                                     RemoteFormatShootingModeADep,
                                     RemoteFormatShootingModeMDep,
                                     RemoteFormatShootingModeBulb,
                                     RemoteFormatShootingModeManual2,
                                     RemoteFormatShootingModeFlashOff,
                                     RemoteFormatShootingModeLongShutter,
                                     RemoteFormatShootingModeSuperMacro,
                                     RemoteFormatShootingModeFoliage,
                                     RemoteFormatShootingModeIndoor,
                                     RemoteFormatShootingModeFireworks,
                                     RemoteFormatShootingModeBeach,
                                     RemoteFormatShootingModeUnderwater,
                                     RemoteFormatShootingModeSnow,
                                     RemoteFormatShootingModeKidsAndPets,
                                     RemoteFormatShootingModeNightSnapshot,
                                     RemoteFormatShootingModeDigitalMacro,
                                     RemoteFormatShootingModeMyColors,
                                     RemoteFormatShootingModePhotoInMovie);

//Remote drive mode
type RemoteFormatDriveModeType = (RemoteFormatDriveModeNotUsed,
                                  RemoteFormatDriveModeSingleFrame,
                                  RemoteFormatDriveModeContinous,
                                  RemoteFormatDriveModeVideo,
                                  RemoteFormatDriveModeHighSpeedContinous,
                                  RemoteFormatDriveModeLowSpeedContinous,
                                  RemoteFormatDriveMode10SecSelfTimer,
                                  RemoteFormatDriveMode2SecSelfTimer);

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

//Remote White Balance
type RemoteFormatWBType = (RemoteFormatWBNotUsed,
                           RemoteFormatWBAuto,
                           RemoteFormatWBDaylight,
                           RemoteFormatWBCloudy,
                           RemoteFormatWBTungsten,
                           RemoteFormatWBFluorscent,
                           RemoteFormatWBFlash,
                           RemoteFormatWBFluorescentLight,
                           RemoteFormatWBCustom,
                           RemoteFormatWBCustom1,
                           RemoteFormatWBCustom2,
                           RemoteFormatWBBW,
                           RemoteFormatWBShade,
                           RemoteFormatWBKelvin,
                           RemoteFormatWBPCSet1,
                           RemoteFormatWBPCSet2,
                           RemoteFormatWBPCSet3);

//Remote Autofocus Mode
type RemoteFormatAFModeType = (RemoteFormatAFModeNotUsed,
                               RemoteFormatAFModeOneShot,
                               RemoteFormatAFModeAIServo,
                               RemoteFormatAFModeAIFocus,
                               RemoteFormatAFModeManual);

//Remote Autofocus Distance
type RemoteFormatAFDistType = (RemoteFormatAFDistNotUsed,
                               RemoteFormatAFDistManual,
                               RemoteFormatAFDistAuto,
                               RemoteFormatAFDistUnknown,
                               RemoteFormatAFDistZFCloseUp,
                               RemoteFormatAFDistZFShortestDistance,
                               RemoteFormatAFDistZFShortDistance,
                               RemoteFormatAFDistZFMediumDistance,
                               RemoteFormatAFDistZFFarDistance,
                               RemoteFormatAFDistPanFocus,
                               RemoteFormatAFDistSuperMacro,
                               RemoteFormatAFDistInfinity,
                               RemoteFormatAFDistSuperMacroOCM);

//Remote Autofocus Focusing Point(s)
type RemoteFormatAFFocusingPointType = (RemoteFormatAFFocusingPointNotUsed,
                                        RemoteFormatAFFocusingPointOnCenterOnlyManual,
                                        RemoteFormatAFFocusingPointOnCenterOnlyAuto,
                                        RemoteFormatAFFocusingPointMultipleManual,
                                        RemoteFormatAFFocusingPointMultipleAuto,
                                        RemoteFormatAFFocusingPointMultipleRight,
                                        RemoteFormatAFFocusingPointMultipleCenter,
                                        RemoteFormatAFFocusingPointMultipleLeft);

//Remote AFLight
type RemoteFormatAFLightType = (RemoteFormatAFLightNotUsed,
                                RemoteFormatAFLightOn,
                                RemoteFormatAFLightOff);

//Remote Flash setting
type RemoteFormatFlashType = (RemoteFormatFlashNotUsed,
                              RemoteFormatFlashOff,
                              RemoteFormatFlashAuto,
                              RemoteFormatFlashOn,
                              RemoteFormatFlashRedEye,
                              RemoteFormatFlashSlowSync,
                              RemoteFormatFlashAutoRedEye,
                              RemoteFormatFlashOnRedEye);

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

//Remote ML Weighting
type RemoteFormatMLWeiType = (RemoteFormatMLWeiNotUsed,
                              RemoteFormatMLWeiCenterWeighted,
                              RemoteFormatMLWeiSpot,
                              RemoteFormatMLWeiAveraging,
                              RemoteFormatMLWeiEvaluative,
                              RemoteFormatMLWeiPartial,
                              RemoteFormatMLWeiCenterwWeightedAveraging);

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

//Remote ColorSpace
type RemoteFormatColorSpaceType = (RemoteFormatColorSpaceNotUsed,
                                   RemoteFormatColorSpaceSRGB,
                                   RemoteFormatColorSpaceAdobeRGB);

//Remote ISO
type RemoteFormatISOType = (RemoteFormatISONotUsed,
                            RemoteFormatISOAuto,
                            RemoteFormatISO50,
                            RemoteFormatISO64,
                            RemoteFormatISO80,
                            RemoteFormatISO100,
                            RemoteFormatISO125,
                            RemoteFormatISO160,
                            RemoteFormatISO200,
                            RemoteFormatISO250,
                            RemoteFormatISO320,
                            RemoteFormatISO400,
                            RemoteFormatISO500,
                            RemoteFormatISO640,
                            RemoteFormatISO800,
                            RemoteFormatISO1000,
                            RemoteFormatISO1250,
                            RemoteFormatISO1600,
                            RemoteFormatISO2000,
                            RemoteFormatISO2500,
                            RemoteFormatISO3200);

//Remote TV, in 1/3 steps
type RemoteFormatTVType = (RemoteFormatTVNotUsed,
                           RemoteFormatTV30Bulb,
                           RemoteFormatTV30sec,
                           RemoteFormatTV25sec,
                           RemoteFormatTV20sec,
                           RemoteFormatTV15sec,
                           RemoteFormatTV13sec,
                           RemoteFormatTV10sec,
                           RemoteFormatTV8sec,
                           RemoteFormatTV6sec,
                           RemoteFormatTV5sec,
                           RemoteFormatTV4sec,
                           RemoteFormatTV3sec2,
                           RemoteFormatTV2sec5,
                           RemoteFormatTV2sec,
                           RemoteFormatTV1sec6,
                           RemoteFormatTV1sec3,
                           RemoteFormatTV1sec,
                           RemoteFormatTV0sec8,
                           RemoteFormatTV0sec6,
                           RemoteFormatTV0sec5,
                           RemoteFormatTV0sec4,
                           RemoteFormatTV0sec3,
                           RemoteFormatTV1_4,
                           RemoteFormatTV1_5,
                           RemoteFormatTV1_6,
                           RemoteFormatTV1_8,
                           RemoteFormatTV1_10,
                           RemoteFormatTV1_13,
                           RemoteFormatTV1_15,
                           RemoteFormatTV1_20,
                           RemoteFormatTV1_25,
                           RemoteFormatTV1_30,
                           RemoteFormatTV1_40,
                           RemoteFormatTV1_50,
                           RemoteFormatTV1_60,
                           RemoteFormatTV1_80,
                           RemoteFormatTV1_100,
                           RemoteFormatTV1_125,
                           RemoteFormatTV1_160,
                           RemoteFormatTV1_200,
                           RemoteFormatTV1_250,
                           RemoteFormatTV1_320,
                           RemoteFormatTV1_400,
                           RemoteFormatTV1_500,
                           RemoteFormatTV1_640,
                           RemoteFormatTV1_800,
                           RemoteFormatTV1_1000,
                           RemoteFormatTV1_1250,
                           RemoteFormatTV1_1600,
                           RemoteFormatTV1_2000,
                           RemoteFormatTV1_2500,
                           RemoteFormatTV1_3200,
                           RemoteFormatTV1_4000,
                           RemoteFormatTV1_5000,
                           RemoteFormatTV1_6400,
                           RemoteFormatTV1_8000,
                           RemoteFormatTV1_10000,
                           RemoteFormatTV1_12800,
                           RemoteFormatTV1_16000);

//Remote AV, in 1/3 steps
type RemoteFormatAVType = (RemoteFormatAVNotUsed,
                           RemoteFormatAV1_0,
                           RemoteFormatAV1_1,
                           RemoteFormatAV1_2,
                           RemoteFormatAV1_4,
                           RemoteFormatAV1_6,
                           RemoteFormatAV1_8,
                           RemoteFormatAV2_0,
                           RemoteFormatAV2_2,
                           RemoteFormatAV2_5,
                           RemoteFormatAV2_8,
                           RemoteFormatAV3_2,
                           RemoteFormatAV3_5,
                           RemoteFormatAV4_0,
                           RemoteFormatAV4_5,
                           RemoteFormatAV5_0,
                           RemoteFormatAV5_6,
                           RemoteFormatAV6_3,
                           RemoteFormatAV7_1,
                           RemoteFormatAV8_0,
                           RemoteFormatAV9_0,
                           RemoteFormatAV10_0,
                           RemoteFormatAV11_0,
                           RemoteFormatAV13_0,
                           RemoteFormatAV14_0,
                           RemoteFormatAV16_0,
                           RemoteFormatAV18_0,
                           RemoteFormatAV20_0,
                           RemoteFormatAV22_0,
                           RemoteFormatAV25_0,
                           RemoteFormatAV29_0,
                           RemoteFormatAV32_0,
                           RemoteFormatAV36_0,
                           RemoteFormatAV40_0,
                           RemoteFormatAV45_0,
                           RemoteFormatAV51_0,
                           RemoteFormatAV57_0,
                           RemoteFormatAV64_0,
                           RemoteFormatAV72_0,
                           RemoteFormatAV81_0,
                           RemoteFormatAV91_0);

//Remote Photo effect
type RemoteFormatPhotoEffectType = (RemoteFormatPhotoEffectNotUsed,
                                    RemoteFormatPhotoEffectOff,
                                    RemoteFormatPhotoEffectVivid,
                                    RemoteFormatPhotoEffectNeutral,
                                    RemoteFormatPhotoEffectLowSharpening,
                                    RemoteFormatPhotoEffectSepia,
                                    RemoteFormatPhotoEffectBW);

//Remote Beep
type RemoteFormatBeepType = (RemoteFormatBeepNotUsed,
                             RemoteFormatBeepOn,
                             RemoteFormatBeepOff);

//PictureStyle. Only supported by 5D, 30D and newer EOS cameras in EOS-DLL
type Int0_7Type  = 0..7;
     IntN4_4Type = -4..4;
     MonochromeFilterType = (MonochromeFilterNone,
                             MonochromeFilterYellow,
                             MonochromeFilterOrange,
                             MonochromeFilterRed,
                             MonochromeFilterGreen);
     MonochromeToneType = (MonochromeToneNone,
                           MonochromeToneSepia,
                           MonochromeToneBlue,
                           MonochromeToneViolet,
                           MonochromeToneGreen);
     //Either use "Saturation + ColorTone" or "MonochromeFilter + MonochromeTone"
     //for black and white picture styles. Please set the "Prop"Used flags correct, e.g.
     //SaturationUsed, ColorToneUsed := true
     //MonochromeFilterUsed, MonochromeToneUsed := false
     PictureStyleType = record
                          PictureStyleEnabled  : boolean;
                          Contrast             : IntN4_4Type;
                          Sharpness            : Int0_7Type;
                          Saturation           : IntN4_4Type;
                          SaturationUsed       : boolean;
                          ColorTone            : IntN4_4Type;
                          ColorToneUsed        : boolean;
                          MonochromeFilter     : MonochromeFilterType;
                          MonochromeFilterUsed : boolean;
                          MonochromeTone       : MonochromeToneType;
                          MonochromeToneUsed   : boolean;
                        end;
//WhitebalanceShift
```

```
type IntN9_9Type  = -9..9;
     RemoteWhitebalanceShiftType = record
                                     WBShiftEnabled : boolean;
                                     AmberBlue      : IntN9_9Type;
                                     GreenMagenta   : IntN9_9Type;
                                   end;
```

Description
Type used to get and set remote parameters to/from the camera. Is used by the RemoteSetRemoteParams 24 and RemoteGetRemoteParams 19 methods.
The CompQualityPic2RAW and ImageSizePic2RAW parameters are used to set the JPEG picture quality and image size, when taking a RAW+JPEG picture. CompQuality must therefore be set to RemoteFormatQualityRAW. Set CompQualityPic2RAW and ImageSizePic2RAW properties to "not used" if only a RAW picture without an added JPEG picture is wanted. CompQualityPic2RAW and ImageSizePic2RAW is not used when a JPEG picture without RAW data is taken.
The drivemode parameter is not probed correctly with at least an EOS 5D and EOS-DLL. All elements in the drivemode seems to be accepted, however only a few really works with the camera. To avoid problems, please use only the working elements.

3.6.7 TRAWParamType

Syntax:

```
TRAWParamType = (RAWParamWhiteBalanceSetting,
                 RAWParamWhiteBalanceKelvin);
```

3.6.8 TRemoteParamType

Syntax:

```
TRemoteParamType = (RemoteParamCompQuality,
                    RemoteParamCompQualityPic2RAW,
                    RemoteParamImageSize,
                    RemoteParamImageSizePic2RAW,
                    RemoteParamStrobeSetting,
                    RemoteParamStrobeCompSetting,
                    RemoteParamDriveMode,
                    RemoteParamImageMode,
                    RemoteParamMLWeiMode,
                    RemoteParamAFMode,
                    RemoteParamAFDistance,
                    RemoteParamAFFocusingPoint,
                    RemoteParamAFAssistLight,
                    RemoteParamWhiteBalanceSetting,
                    RemoteParamWhiteBalanceKelvin,
                    RemoteParamWhiteBalanceShift,
                    RemoteParamPictureStyle,
                    RemoteParamContrast,
                    RemoteParamColorGain,
                    RemoteParamSharpness,
                    RemoteParamColorSpace,
                    RemoteParamISO,
                    RemoteParamAv,
                    RemoteParamTv,
                    RemoteParamExposureCompensation,
                    RemoteParamPhotoEffect,
                    RemoteParamBeep);
```

3.6.9 TRemoteParamPictureStyleType

Syntax:

```
TRemoteParamPictureStyleType = (RemoteParamPictureStyleEnabled,
                                RemoteParamPictureStyleContrast,
                                RemoteParamPictureStyleSharpness,
                                RemoteParamPictureStyleSaturation,
                                RemoteParamPictureStyleSaturationUsed,
                                RemoteParamPictureStyleColorTone,
                                RemoteParamPictureStyleColorToneUsed,
                                RemoteParamPictureStyleMonochromeFilter,
                                RemoteParamPictureStyleMonochromeFilterUsed,
                                RemoteParamPictureStyleMonochromeTone,
                                RemoteParamPictureStyleMonochromeToneUsed);
```

3.6.10 TRemoteParamWhitebalanceShiftType

Syntax:

```
TRemoteParamWhitebalanceShiftType = (RemoteParamWhitebalanceShiftWBShiftEnabled,
                                     RemoteParamWhitebalanceShiftAmberBlue,
                                     RemoteParamWhitebalanceShiftGreenMagenta);
```

Index

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.