|  |  |  |
| --- | --- | --- |
| **TrueTestEngine .NET API Specification** | |  |
| Version: | 1.3 |  |
| Subject: | *.NET API specifications for TrueTestEngine version 1.2.8 or higher.* | Research and Development Application Note |
| Contents: | 89 Pages, No Disks | March, 2018 |

Running the Microsoft Mail 3.2 Windows Client from the Network

WA0888

8/25/93

1993

**Application Note History**

Application Note Name: *TrueTestEngine .NET API.doc*

Title: *TrueTestEngine .NET API Specifications*

Product: TrueTestEngine.dll, TrueTestPatternGenerator.dll

Versions: TrueTestEngine 1.2.8

Original Author: SSkelly

Creation Date: April, 2012

Release Date: January, 2013

Updates

|  |  |
| --- | --- |
| Revision Date | Changes by/ Comments |
| April, 2012 | SSkelly / Created version 1.1 |
| January, 2013 | SSkelly / Revision update – new API functions |
| April, 2013 | SSkelly / Revision update – namespace change, new functions |
| March, 2015 | AUsher/ Revision update – company name change |
| March, 2018 | KBilty/ Revision update – updates |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

*.*

INFORMATION PROVIDED IN THIS DOCUMENT AND ANY SOFTWARE THAT MAY ACCOMPANY THIS DOCUMENT (collectively referred to as a Application Note) IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR PURPOSE. The user assumes the entire risk as to the accuracy and the use of this Application Note.   
  
 Copyright © 1993-2018 Radiant Vision Systems, LLC. All Rights Reserved.  
 Windows and Word are registered trademarks of Microsoft Corporation.

ProMetric is a registered trademark of Radiant Vision Systems, LLC.  
 *This document was created using Microsoft Word for Windows.*

**Table of Contents**

[INTRODUCTION 14](#_Toc507664073)

[Initial Setup (Analysis): 14](#_Toc507664074)

[Initial Setup (Pattern, Pattern Generator): 14](#_Toc507664075)

[Initial Setup (User Interface): 14](#_Toc507664076)

[TrueTest Class 15](#_Toc507664077)

[Background 15](#_Toc507664078)

[Public Methods 15](#_Toc507664079)

[AppSettings Method 15](#_Toc507664080)

[Retrieves the current set of global Settings.](#_Toc507664081)

[CameraCalibrationDatabaseSetWithBrowse Method 15](#_Toc507664082)

[Lets the user browse to set a \*.calx camera calibration database file with the desired filename. The filename the user selected is returned via the argument CalibrationDBpath. This requires a restart of the software, as changing the calibration database also implies changing the camera.](#_Toc507664083)

[ClearMeasurementHistory Method 15](#_Toc507664084)

[Clears the measurement history saved in the current measurement database. This deletes all generated measurements (Local Contrast, RADA, Reference, FFT, etc), and deletes any Results. In effect, it cleans the database of all data except original camera measurements. Optionally you can clear the measurement history for an individual serial number.](#_Toc507664085)

[Initialize Method 16](#_Toc507664086)

[Initializes all components of TrueTest. Call this during the load process of the application. Requires input of three database file paths to initialize.](#_Toc507664087)

[InitializeAppSettings Method 16](#_Toc507664088)

[Resets the global Settings class to the most recent Settings that were saved.](#_Toc507664089)

[InitializeCamera Method 16](#_Toc507664090)

[Connects to and initializes a camera using the desired camera calibration database.](#_Toc507664091)

[MeasurementDatabaseNew Method 17](#_Toc507664092)

[Creates a new \*.ttxm measurement database file with the desired filename.](#_Toc507664093)

[MeasurementDatabaseNewWithBrowse Method 17](#_Toc507664094)

[Lets the user browse to creates a new \*.ttxm measurement database file with the desired filename. The filename is returned via the argument MeasurementDBpath.](#_Toc507664095)

[MeasurementDatabaseSet Method 18](#_Toc507664096)

[Sets the \*.ttxm measurement database file to the desired filename.](#_Toc507664097)

[MeasurementDatabaseSetWithBrowse Method 18](#_Toc507664098)

[Lets the user browse to set a \*.ttxm measurement database file with the desired filename. The filename the user selected is returned via the argument MeasurementDBpath.](#_Toc507664099)

[MeasurementDatabaseSetWithLoop Method 18](#_Toc507664100)

[Lets the user browse to set a \*.ttxm measurement database file with the desired filename. The filename the user selected is returned via the argument MeasurementDBpath. If the initial argument for MeasurementDBpath cannot be found, the user is given a choice of browsing to select a database or browsing to select a path to create a new database.](#_Toc507664101)

[SequenceNew Method 19](#_Toc507664102)

[Creates a new \*.seqx Sequence file with the desired filename.](#_Toc507664103)

[SequenceNewWithBrowse Method 19](#_Toc507664104)

[Lets the user browse to select a new \*.seqx Sequence file, starting in the folder containing the desired filename. The filename is returned By Reference in the argument SequenceDBpath.](#_Toc507664105)

[SequenceRunAll Method 19](#_Toc507664106)

[Runs all steps in the current sequence from the beginning. Default run mode is Synchronous.](#_Toc507664107)

[SequenceSave Method 19](#_Toc507664108)

[Saves the current sequence to the file path SetupXMLFilePathName. Equivalent to File->Save As sequence. Also equivalent to File->Save if the file path name already exists on disk (overwrites).](#_Toc507664109)

[SequenceSaveRequired Method 20](#_Toc507664110)

[Returns whether or not the sequence has been edited since its last save.](#_Toc507664111)

[SequenceSet Method 20](#_Toc507664112)

[Sets the current sequence using the \*.seqx file found at SetupXMLFilePathName. Equivalent to File->Open sequence.](#_Toc507664113)

[SequenceSetWithBrowse Method 20](#_Toc507664114)

[Sets the current sequence using the \*.seqx file found at SequenceDBpath. Equivalent to File->Open sequence. If no file is found at SequenceDBpath, then the user is presented with a browse dialog.](#_Toc507664115)

[SequenceSetWithLoop Method 21](#_Toc507664116)

[Sets the current sequence using the \*.seqx file found at SequenceDBpath. Equivalent to File->Open sequence. If no file is found at SequenceDBpath, then the user is presented with a browse dialog. If the user tries to cancel the dialog, they are asked if they want to try again or browse to create a new sequence.](#_Toc507664117)

[SequenceStop Method 21](#_Toc507664118)

[Stops the current sequence. If the user wants to cancel, call this method.](#_Toc507664119)

[SetPatternReady Method 21](#_Toc507664120)

[Calls the ‘SetPatternReady’ method of the Pattern generator. Used if the pattern generator is designed to wait for this response before continuing.](#_Toc507664121)

[ShowEditSequenceDialog Method 21](#_Toc507664122)

[Shows a dialog allowing the user to edit the sequence. Use only if you are creating a custom UI and want to easily control the ability to edit a sequence.](#_Toc507664123)

[ShowPatternGeneratorControlForm Method 22](#_Toc507664124)

[Shows a dialog allowing the user to edit pattern generator settings. Only shows a dialog if the PatternGenerator property HasCustomDeviceControlForm is set to True, and the method GetCustomForm was overridden properly.](#_Toc507664125)

[ShowSerialNumberDialog Method 22](#_Toc507664126)

[Shows a dialog that allows the user to enter a serial number. Includes some optional parameters for showing and saving the serial number.](#_Toc507664127)

[Shutdown Method 22](#_Toc507664128)

[Shutsdown the camera and clears TrueTest settings.](#_Toc507664129)

[TakeMeasurement Method 22](#_Toc507664130)

[Shows a dialog allowing the user to select a PatternSetup and take a manual measurement. The measurement will be saved as an original camera measurement into the measurement database, but will not be sent to an Analysis.](#_Toc507664131)

[TakeMeasurement Method 23](#_Toc507664132)

[Takes a measurement using the PatternSetup provided. The measurement will be returned and also saved as an original camera measurement, but will not be sent to an Analysis.](#_Toc507664133)

[UIForm Method 23](#_Toc507664134)

[Returns the user interface form connected to TrueTest.](#_Toc507664135)

[Public Properties 24](#_Toc507664136)

[CalibrationFilePathName Property (ReadOnly) 24](#_Toc507664137)

[Gives access to the currently selected Calibration database (\*.calx) where camera calibration information is stored.](#_Toc507664138)

[MeasurementDatabasePathFileName Property (ReadOnly) 24](#_Toc507664139)

[Gives access to the currently selected Measurement database (\*.ttxm) where measurements and results are stored.](#_Toc507664140)

[OperatingMode Property 24](#_Toc507664141)

[Sets the current mode of operation: Use Camera, Database (Auto), Database (User Select). Database modes are most often used for troubleshooting or testing.](#_Toc507664142)

[PatternGenerator Property 24](#_Toc507664143)

[Gives access to the current Pattern Generator object used to generate patterns to be measured on the device under test. The Pattern Generator can also include any external hardware control commands.](#_Toc507664144)

[PMEngine Property 25](#_Toc507664145)

[Gives access to PMEngine for direct control of the camera.](#_Toc507664146)

[SaveMeasurementsToDatabase Property 25](#_Toc507664147)

[Sets whether or not to save camera measurements to the database.](#_Toc507664148)

[Sequence Property 25](#_Toc507664149)

[Gives access to the current Sequence, which contains all the setup information for taking and analyzing measurements. Each Sequence is saved in a separate sequence file with the \*.seqx extension.](#_Toc507664150)

[SerialNumber Property 25](#_Toc507664151)

[Gives access to the most recently entered Serial Number of a device under test.](#_Toc507664152)

[UserDataFolder Property 26](#_Toc507664153)

[Gets the User Data Folder.](#_Toc507664154)

[Public Events 26](#_Toc507664155)

[\_AnalysisComplete Event 26](#_Toc507664156)

[Occurs when an analysis has completed.](#_Toc507664157)

[\_DefectListChanged Event 26](#_Toc507664158)

[Occurs when an analysis is requesting to show a defect list to the user.](#_Toc507664159)

[\_MeasurementChanged Event 27](#_Toc507664160)

[Occurs when a measurement has had additional processing, and is requested to be shown to the user.](#_Toc507664161)

[\_MeasurementComplete Event 27](#_Toc507664162)

[Occurs when a camera measurement has been fully completed and is about to be passed to the Analysis.](#_Toc507664163)

[\_ROIArrayChanged Event 27](#_Toc507664164)

[Occurs when an anlaysis is requesting to show an ROI array to the user.](#_Toc507664165)

[\_ShowMeasurement Event 28](#_Toc507664166)

[Occurs when an analysis is requesting to show a measurement to the user.](#_Toc507664167)

[\_ShowCustomForm Event 28](#_Toc507664168)

[Occurs when an analysis wishes to show a Custom form (different from the standard bitmap form) to the user.](#_Toc507664169)

[\_UpdateCustomForm Event 28](#_Toc507664170)

[Occurs when an analysis wishes to update a Custom form (different from the standard bitmap form) already shown to the user.](#_Toc507664171)

[ExposureComplete Event 29](#_Toc507664172)

[Occurs every time an exposure within a camera measurement has been completed.](#_Toc507664173)

[MeasurementDatabaseChanged Event 29](#_Toc507664174)

[Occurs when the user has selected a new measurement database file (\*.ttxm).](#_Toc507664175)

[SequenceChanged Event 29](#_Toc507664176)

[Occurs when the user has selected a new sequence file (\*.seqx).](#_Toc507664177)

[SequenceComplete Event 29](#_Toc507664178)

[Occurs when a sequence has completed.](#_Toc507664179)

[ShowPatternCommandSent Event 30](#_Toc507664180)

[Occurs when the Pattern Generator is trying to Show a Pattern.](#_Toc507664181)

[AnalysisBase Class 31](#_Toc507664182)

[Background 31](#_Toc507664183)

[MustOverride Methods and Properties 31](#_Toc507664184)

[Execute Method 31](#_Toc507664185)

[Executes the code that performs the analysis. Override this function and put image processing code here.](#_Toc507664186)

[Name Property (ReadOnly) 31](#_Toc507664187)

[Sets the text Name that will be written to the Results database for results of the Analysis.](#_Toc507664188)

[UserName Property 31](#_Toc507664189)

[Sets the initial User Name that the user gets to edit.](#_Toc507664190)

[Protected Properties 31](#_Toc507664191)

[SequenceIndex Property 31](#_Toc507664192)

[The current index of the analysis within the Sequence class list Items. Can be used to know which step in the sequence is currently executing.](#_Toc507664193)

[SecondSequenceIndex Property 32](#_Toc507664194)

[A secondary index. Used if an Analysis wishes to maintain multiple bitmap windows.](#_Toc507664195)

[PassFail Property 32](#_Toc507664196)

[Shows whether or not the analysis has passed or failed.](#_Toc507664197)

[PatternNameList Property 32](#_Toc507664198)

[Contains a list of strings corresponding to the names of the patterns used to generate the measurements being passed in to this Analysis.](#_Toc507664199)

[PatternMeasurementList Property 32](#_Toc507664200)

[Contains a list of measurements for this Analysis to execute with. The list type is of class PatternMeasurement.](#_Toc507664201)

[ResultList Property 33](#_Toc507664202)

[Contains a list of Result objects to be written to the database and potentially used to generate a report.](#_Toc507664203)

[ROIArray Property 33](#_Toc507664204)

[Contains a list of ROI objects.](#_Toc507664205)

[Protected Methods 33](#_Toc507664206)

[Deserialize Method 33](#_Toc507664207)

[Used to deserialize objects from XML strings when saving custom properties of an Analysis.](#_Toc507664208)

[OnMeasurementChanged Method 34](#_Toc507664209)

[Used to raise MeasurementChanged event from within an Execute function.](#_Toc507664210)

[OnROIArrayChanged Method 34](#_Toc507664211)

[Used to raise ROIArrayChanged event from within an Execute function.](#_Toc507664212)

[Serialize Method 34](#_Toc507664213)

[Used to serialize objects to XML strings when saving custom properties of an Analysis.](#_Toc507664214)

[SetAnalysisLevelPassFail Method 34](#_Toc507664215)

[Sets the PassFail property of the AnalysisBase based on the pass/fail values inside the AnalysisBase property ResultList. Can be overridden if the programmer wishes to ignore certain results, or set the PassFail property manually. This method is called automatically by TrueTestEngine during runtime.](#_Toc507664216)

[WritePMMeasurementToDatabase Method 35](#_Toc507664217)

[Writes a RadiantCommon.MeasurementF object to the measurement database (\*.ttxm file) for further manual analysis.](#_Toc507664218)

[EventArgs Classes 36](#_Toc507664219)

[Background 36](#_Toc507664220)

[AnalysisCompleteEventArgs 36](#_Toc507664221)

[AnalysisUserName Property 36](#_Toc507664222)

[Contains the text name of the Analysis that was completed.](#_Toc507664223)

[PassFail Property 36](#_Toc507664224)

[Reports whether the Analysis passed or failed.](#_Toc507664225)

[PatternName Property 36](#_Toc507664226)

[Reports the pattern name.](#_Toc507664227)

[Results Property 36](#_Toc507664228)

[Contains a list of Results from the Analysis that has completed.](#_Toc507664229)

[SequenceIndex Property 37](#_Toc507664230)

[The index number of the analysis which has completed.](#_Toc507664231)

[DefectListChangedEventArgs 37](#_Toc507664232)

[AppendToList Property 37](#_Toc507664233)

[Reports whether or not the Defect List should be appended to an existing list.](#_Toc507664234)

[BlobDrawShape Property 37](#_Toc507664235)

[Reports the shape to draw if blobs are found inside the Defect List.](#_Toc507664236)

[DefectList Property 37](#_Toc507664237)

[The list of Defects. The purpose of this list is to get drawn on a RadiantCommon.RiBitmapCtl to show the user where defects were found.](#_Toc507664238)

[Measurement Property 38](#_Toc507664239)

[A camera measurement object related to the defects in DefectList.](#_Toc507664240)

[SequenceIndex Property 38](#_Toc507664241)

[Reports the index of the analysis in the sequence that the DefectList came from.](#_Toc507664242)

[SecondSequenceIndex Property 38](#_Toc507664243)

[Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.](#_Toc507664244)

[ExposureCompleteEventArgs 38](#_Toc507664245)

[SequenceIndex Property 38](#_Toc507664246)

[Reports the index of the analysis in the sequence that the DefectList came from.](#_Toc507664247)

[NextPattern Property 38](#_Toc507664248)

[Contains the PatternBase object that the sequence wants to be shown before taking the next camera measurement.](#_Toc507664249)

[MeasurementCompleteEventArgs 39](#_Toc507664250)

[Measurement Property 39](#_Toc507664251)

[A camera measurement object returned when the camera has completed taking a measurement.](#_Toc507664252)

[PatternSetup Property 39](#_Toc507664253)

[The collection of properties used to take the measurement which has completed.](#_Toc507664254)

[SequenceIndex Property 39](#_Toc507664255)

[Reports the index of the analysis in the sequence that the Measurement came from.](#_Toc507664256)

[SecondSequenceIndex Property 39](#_Toc507664257)

[Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.](#_Toc507664258)

[ROIArrayChangedEventArgs 40](#_Toc507664259)

[ROIArray Property 40](#_Toc507664260)

[An array of Regions Of Interest (ROIs) to be displayed to the user. ROIs can be circles, rectangles, ellipses, or polygons.](#_Toc507664261)

[SequenceIndex Property 40](#_Toc507664262)

[Reports the index of the analysis in the sequence that the ROIArray came from.](#_Toc507664263)

[SecondSequenceIndex Property 40](#_Toc507664264)

[Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.](#_Toc507664265)

[SequenceCompleteEventsArgs 40](#_Toc507664266)

[Canceled Property 40](#_Toc507664267)

[Reports if the sequence has been canceled.](#_Toc507664268)

[NumFailedTests Property 41](#_Toc507664269)

[Reports the number of tests in the completed sequence that have failed.](#_Toc507664270)

[PassFail Property 41](#_Toc507664271)

[Reports whether or not the entire sequence passed or failed.](#_Toc507664272)

[SequenceName Property 41](#_Toc507664273)

[Reports the name of the completed sequence.](#_Toc507664274)

[SerialNumber Property 41](#_Toc507664275)

[Reports the serial number of the DUT for the completed sequence.](#_Toc507664276)

[ObjectRepository Class 42](#_Toc507664277)

[Background 42](#_Toc507664278)

[Public Enums 42](#_Toc507664279)

[ReadWriteEnum 42](#_Toc507664280)

[Used with GetItem. Sets whether or not code will only try to Read from the Repository, or will Write if the requested item wasn’t found.](#_Toc507664281)

[Public Methods 42](#_Toc507664282)

[AddItem Method 42](#_Toc507664283)

[Adds an item to the repository, using the specified key.](#_Toc507664284)

[GetItem Method 42](#_Toc507664285)

[Gets an item from the repository, using the specified key.](#_Toc507664286)

[PatternMeasurement Class 44](#_Toc507664287)

[Background 44](#_Toc507664288)

[Public Properties 44](#_Toc507664289)

[CurrentMeasurement Property 44](#_Toc507664290)

[The current index of the analysis within the Sequence class list Items. Can be used to know which step in the sequence is currently executing.](#_Toc507664291)

[CameraMeasurementID Property (Read Only) 44](#_Toc507664292)

[Contains the unique ID number of the measurement in the database (\*.ttxm). If this is a new camera measurement that has not yet been written to the database, 0 will be returned.](#_Toc507664293)

[BlobDrawShape Property 44](#_Toc507664294)

[Selects a shape to draw for any blobs inside the DefectList.](#_Toc507664295)

[DefectList Property 45](#_Toc507664296)

[Contains a list of defect objects to be shown to the user.](#_Toc507664297)

[LocalContrast Property 45](#_Toc507664298)

[Contains a processed Local Contrast measurement based on the original camera measurement.](#_Toc507664299)

[PatternSetup Property 45](#_Toc507664300)

[The PatternSetup object used to take the original camera measurement.](#_Toc507664301)

[RADA Property 45](#_Toc507664302)

[RADA stands for Register Active Display Area. For a rectangular device under test, a RADA algorithm can be applied that Registers the location of the active display area within the camera’s field of view. If this registration process has been performed, a rectangular measurement object is created removing any inactive border around the device under test, and saved to this property.](#_Toc507664303)

[Reference Property 46](#_Toc507664304)

[In some defect processing Analyses, the original camera measurement is compared to a Reference image created from the original measurement to highlight defects. This property stores that Reference image for later review.](#_Toc507664305)

[ROIArray Property 46](#_Toc507664306)

[An array of Regions Of Interest (ROIs) to be displayed to the user. ROIs can be circles, rectangles, ellipses, or polygons.](#_Toc507664307)

[SerialNumber Property 46](#_Toc507664308)

[The serial number of the unit whose images and data are contained in this PatternMeasurement.](#_Toc507664309)

[SyntheticMeasurement Property 46](#_Toc507664310)

[A synthetic measurement generated from data inside the original measurement. This measurement is most often a downsampling of data from the original measurement in a grid pattern, such as averaging data from a grid of Regions Of Interest.](#_Toc507664311)

[PatternSetup Class 48](#_Toc507664312)

[Background 48](#_Toc507664313)

[Constructors 48](#_Toc507664314)

[New 48](#_Toc507664315)

[Initializes a new instance of the PatternSetup class.](#_Toc507664316)

[Public Methods 48](#_Toc507664317)

[CloneMe Method 48](#_Toc507664318)

[Clones the instance of the PatternSetup.](#_Toc507664319)

[GetMeasurementSetup Method 48](#_Toc507664320)

[Gets the measurement setup object. Used if you want to take measurements through the PMEngine API instead of TrueTestEngine API. Note that TrueTest saves some additional metadata that PMEngine is unaware of. For the TrueTest Take Measurement function, see the TrueTest class.](#_Toc507664321)

[SetFromMeasurementSetup Method 49](#_Toc507664322)

[Sets the properties of a PatternSetup object based on a PMEngine.MeasurementSetup object.](#_Toc507664323)

[Public Properties 49](#_Toc507664324)

[ApplyMedianSmoothing Property 49](#_Toc507664325)

[True or False whether to apply a median smoothing filter after taking the measurement.](#_Toc507664326)

[AutoAdjustExposure Property 49](#_Toc507664327)

[True or False whether to automatically adjust the exposure of the camera. If false, the software will use fixed exposure times.](#_Toc507664328)

[BinningX Property 49](#_Toc507664329)

[Gets or sets the binning in the x direction (horizontal binning).](#_Toc507664330)

[BinningY Property 50](#_Toc507664331)

[Gets or sets the binning in the y direction (vertical binning).](#_Toc507664332)

[CameraFlip Property 50](#_Toc507664333)

[Gets or sets whether the measurement will be flipped.](#_Toc507664334)

[CameraMirror Property 50](#_Toc507664335)

[Gets or sets whether the measurement will be mirrored.](#_Toc507664336)

[CameraRotation Property 50](#_Toc507664337)

[Gets or sets whether the measurement will be rotated.](#_Toc507664338)

[CameraTranspose Property 51](#_Toc507664339)

[Gets or sets whether the measurement will be transposed.](#_Toc507664340)

[CaptureFilter Property 51](#_Toc507664341)

[Gets or sets whether or not to capture an image for each camera filter](#_Toc507664342)

[ConoscopeCalibrationID Property 51](#_Toc507664343)

[Gets or sets the conoscope calibration ID.](#_Toc507664344)

[DistanceUnit Property 51](#_Toc507664345)

[Gets or sets the distance unit to be applied to measurements.](#_Toc507664346)

[DUTDistance Property 52](#_Toc507664347)

[Gets or sets the distance of the DUT (device under test) to the camera in meters.](#_Toc507664348)

[ExposureLimitMethod Property 52](#_Toc507664349)

[Gets or sets the method for finding the exposure time](#_Toc507664350)

[ExposureShowOverLimitMessage Property 53](#_Toc507664351)

[Gets or sets whether or not to show a message box if a measurement taken with this setup has overexposed](#_Toc507664352)

[ExposureTime Property 53](#_Toc507664353)

[Gets or sets the exposure time in milliseconds for each camera filter](#_Toc507664354)

[GoniometerType Property 53](#_Toc507664355)

[Gets or sets the goniometer type used for creating a luminous intensity measurement from an illuminance measurement](#_Toc507664356)

[GUID Property 54](#_Toc507664357)

[Gets or sets the unique ID associated with this setup](#_Toc507664358)

[HDRImagingEnabled Property 54](#_Toc507664359)

[Gets or sets whether to use HDR (high dynamic range) imaging.](#_Toc507664360)

[LEDColorCalibration Property 54](#_Toc507664361)

[Gets or sets whether to use the LED color calibration.](#_Toc507664362)

[LensDistance Property 54](#_Toc507664363)

[Gets or sets the focus distance for the lens](#_Toc507664364)

[LensfNumber Property 55](#_Toc507664365)

[Gets or sets the f-number to use on the lens.](#_Toc507664366)

[MakeIntensity Property 55](#_Toc507664367)

[Gets or sets whether or not to transform an illuminance measurement into a luminous intensity measurement.](#_Toc507664368)

[MeasurementSetupID Property 55](#_Toc507664369)

[Gets or sets the measurement setup ID](#_Toc507664370)

[MedianBlockSize Property 55](#_Toc507664371)

[Gets or sets the block size (in pixels) used if performing a median filter operation](#_Toc507664372)

[Name Property 56](#_Toc507664373)

[Gets or sets the pattern setup name. This name is required to be unique among all Pattern Setup names in the current Sequence.](#_Toc507664374)

[NbrCaptureFilters Property (Read Only) 56](#_Toc507664375)

[Gets the number of filters used in the measurement](#_Toc507664376)

[NbrDarkFramesToAverage Property 56](#_Toc507664377)

[Gets or sets the number of dark frames to average](#_Toc507664378)

[NbrFramesToAverage Property 56](#_Toc507664379)

[Gets or sets the number of measurements to average](#_Toc507664380)

[NDFilterPosition Property 56](#_Toc507664381)

[Gets or sets the ND filter position to use for each camera filter](#_Toc507664382)

[Notes Property 57](#_Toc507664383)

[Gets or sets the notes on the pattern setup](#_Toc507664384)

[Pattern Property 57](#_Toc507664385)

[Gets or sets the display pattern used for the measurement](#_Toc507664386)

[PhotometricUnit Property 57](#_Toc507664387)

[Gets or sets the photometric quantity associated with this setup](#_Toc507664388)

[SpectralResponse Property 57](#_Toc507664389)

[Gets or sets the spectral response type for this measurement](#_Toc507664390)

[ThresholdFilter Property 58](#_Toc507664391)

[Gets or sets the threshold to apply (as a %) for each camera filter](#_Toc507664392)

[Result Class 59](#_Toc507664393)

[Background 59](#_Toc507664394)

[Constructors 59](#_Toc507664395)

[New 59](#_Toc507664396)

[Initializes a new instance of the Result class.](#_Toc507664397)

[Public Properties 60](#_Toc507664398)

[AnalysisName Property 60](#_Toc507664399)

[The name of the analysis](#_Toc507664400)

[Name Property 60](#_Toc507664401)

[The name of the result.](#_Toc507664402)

[PassFail Property 60](#_Toc507664403)

[Enumeration indicating whether the result passed or failed](#_Toc507664404)

[PatternName Property 61](#_Toc507664405)

[The name of the pattern used during measurement.](#_Toc507664406)

[SequenceName Property 61](#_Toc507664407)

[The name of the sequence where this result came from.](#_Toc507664408)

[SerialNumber Property 61](#_Toc507664409)

[The DUT (device under test) serial number associated with the result.](#_Toc507664410)

[Unit Property 61](#_Toc507664411)

[The unit of the result value (mm, cd/m2, etc)](#_Toc507664412)

[Value Property 62](#_Toc507664413)

[The value of the result reported numerically](#_Toc507664414)

[ValueString Property 62](#_Toc507664415)

[The value of the result reported as a string](#_Toc507664416)

[Sequence Class 63](#_Toc507664417)

[Background 63](#_Toc507664418)

[Constructors 63](#_Toc507664419)

[New 63](#_Toc507664420)

[Initializes a new instance of the Sequence class.](#_Toc507664421)

[Public Methods 63](#_Toc507664422)

[Clone Method 63](#_Toc507664423)

[Makes a clone copy of the current sequence](#_Toc507664424)

[GetPatternSetupByName Method 63](#_Toc507664425)

[Gets the pattern setup object using the name](#_Toc507664426)

[GetSeqItemByGUID Method 64](#_Toc507664427)

[Gets the full SequenceItem (including Analysis object) in the sequence associated with a string GUID.](#_Toc507664428)

[GetSeqItemByGUID Method 64](#_Toc507664429)

[Gets the full SequenceItem (including Analysis object) in the sequence associated with a string GUID. This overload requires an empty SequenceItem to be passed in by reference.](#_Toc507664430)

[GetSequenceItemByPatternSetupName Method 64](#_Toc507664431)

[Gets the full SequenceItem (including Analysis object) in the sequence associated with the pattern setup name.](#_Toc507664432)

[LoadComboBoxWithPatterns Method 65](#_Toc507664433)

[Gets a list of pattern setups associated with the analyses in the sequence](#_Toc507664434)

[SetPatternSetup Method 65](#_Toc507664435)

[Updates a pattern setup with values from a new pattern setup. The first pattern setup with the same name as the new pattern setup will get replaced.](#_Toc507664436)

[TestTimeSec Method 65](#_Toc507664437)

[Retrieves the total test time (in seconds) of the most recently run sequence, or 0 if no test time is available.](#_Toc507664438)

[Public Properties 65](#_Toc507664439)

[CurrentSequenceIndex Property 65](#_Toc507664440)

[The index of the current analysis being run.](#_Toc507664441)

[Description Property 66](#_Toc507664442)

[The description for the sequence, which is the sequence filename.](#_Toc507664443)

[IsComplete Property 66](#_Toc507664444)

[True or False indicating if the sequence has been completed](#_Toc507664445)

[Items Property 66](#_Toc507664446)

[The heart of a sequence is its Items list, containing the list of steps to be executed.](#_Toc507664447)

[MasterDefectList Property 66](#_Toc507664448)

[A list of all the defects found from all analyses performed in the sequence.](#_Toc507664449)

[Name Property (Read Only) 66](#_Toc507664450)

[The name of the sequence.](#_Toc507664451)

[NumFailedTests Property (Read Only) 67](#_Toc507664452)

[Number of failed tests in the sequence.](#_Toc507664453)

[NumPassedTests Property (Read Only) 67](#_Toc507664454)

[Number of passed tests in the sequence.](#_Toc507664455)

[PassedSecurity Property 67](#_Toc507664456)

[True or False indicating if the software has passed security. Security checks the license code against the key.](#_Toc507664457)

[PassFail Property 67](#_Toc507664458)

[Indicates whether the sequence has passed or failed.](#_Toc507664459)

[PatternGen Property 68](#_Toc507664460)

[The pattern generator used. This property is only used if the global Settings property ReadPGPropertiesFromSeqxFile is set to True. Otherwise it is null (nothing).](#_Toc507664461)

[PatternSetupList Property 68](#_Toc507664462)

[List of available pattern setups in the sequence definition.](#_Toc507664463)

[SequenceStopped Property (Read Only) 68](#_Toc507664464)

[True or False indicating if the sequence has been stopped/cancelled](#_Toc507664465)

[SerialNumber Property 68](#_Toc507664466)

[Serial number of the device under test.](#_Toc507664467)

[XMLFilePathName Property 69](#_Toc507664468)

[Current File path definition for the sequence. Also used to save the sequence.](#_Toc507664469)

[SequenceItem Class 70](#_Toc507664470)

[Background 70](#_Toc507664471)

[Public Properties 70](#_Toc507664472)

[Analysis Property 70](#_Toc507664473)

[Analysis associated with this item in the sequence.](#_Toc507664474)

[GUID Property 70](#_Toc507664475)

[Globally unique identifier associated with this item in the sequence.](#_Toc507664476)

[Measurement Property (Read Only) 70](#_Toc507664477)

[Measurement associated with this item in the sequence](#_Toc507664478)

[PatternSetupName Property 70](#_Toc507664479)

[Name of the pattern setup associated with this item in the sequence](#_Toc507664480)

[Selected Property 71](#_Toc507664481)

[True or False indicating if the item in the sequence has been selected for analysis.](#_Toc507664482)

[Settings Class 72](#_Toc507664483)

[Background 72](#_Toc507664484)

[Constructors 72](#_Toc507664485)

[New 72](#_Toc507664486)

[Initializes a new instance of the Settings class.](#_Toc507664487)

[Public Methods 72](#_Toc507664488)

[Save 72](#_Toc507664489)

[Saves the current settings.](#_Toc507664490)

[Public Properties 72](#_Toc507664491)

[AnalysisSequenceMode Property 72](#_Toc507664492)

[Mode used to run the analyses. Asynchronous mode makes use of the multithreading features of TrueTest.](#_Toc507664493)

[AnalysisTestSelectionInOperatorMode Property 72](#_Toc507664494)

[Allows the ‘Select’ checkbox of the Sequence Control user interface to appear even in Operator Mode. This means operators can choose to ignore certain tests by deselecting them before running the sequence.](#_Toc507664495)

[AutoGenerateDatabase Property 73](#_Toc507664496)

[Selects whether or not to automatically generate a new measurement database (\*.ttxm) under the following conditions: once daily, and if the current database size is greater than 2.5 GB.](#_Toc507664497)

[CheckDuplicateSerialNumber Property 73](#_Toc507664498)

[Checks if the serial number has already been used.](#_Toc507664499)

[DatabaseModeDelay Property 73](#_Toc507664500)

[Delay (in milliseconds) after each measurement is read from the database.](#_Toc507664501)

[DisplayCameraMeasurements Property 73](#_Toc507664502)

[Whether or not to display original camera measurements to the user.](#_Toc507664503)

[DisplayDefectList Property 73](#_Toc507664504)

[Whether or not to display defect lists to the user.](#_Toc507664505)

[DisplayMasterDefectList Property 74](#_Toc507664506)

[Whether or not to display a master defect list to the user after a sequence completes.](#_Toc507664507)

[DisplayUpdatedMeasurements Property 74](#_Toc507664508)

[Whether or not to display the updated (processed) measurements during the test process to the user.](#_Toc507664509)

[EnterUIinOperatorMode Property 74](#_Toc507664510)

[If set to True, TrueTest will automatically start up in Operator Mode. By default (or if this value is set to False), TrueTest will start in Engineering Mode, allowing anyone to make changes.](#_Toc507664511)

[FailColor Property 74](#_Toc507664512)

[Color used to indicate a failed test](#_Toc507664513)

[MeasurementSequenceMode Property 75](#_Toc507664514)

[Mode used to take the measurements. Asynchronous mode makes use of the multithreading features of TrueTest](#_Toc507664515)

[PassColor Property 75](#_Toc507664516)

[Color used to indicate a passed test](#_Toc507664517)

[Password Property 75](#_Toc507664518)

[Password used to enter the engineering view from the operator view](#_Toc507664519)

[PatternGenerator Property 75](#_Toc507664520)

[Pattern generator selected for the sequence.](#_Toc507664521)

[ReadMSfromCalFile Property 76](#_Toc507664522)

[Whether or not to read Measurement Setup information from the camera calibration file, instead of the sequence (\*.seqx) file. Measurement Setup information is always saved to both \*.seqx and \*.calx camera calibration file.](#_Toc507664523)

[ReadPGpropertiesFromSeqx Property 76](#_Toc507664524)

[Whether or not to save/read Pattern Generator properties to the sequence (\*.seqx) file. If False, the pattern generator properties will be saved globally for all sequence files. If True, each sequence file can save its own pattern generator and pattern generator properties.](#_Toc507664525)

[RegressionTest Property 76](#_Toc507664526)

[If set to True, once the sequence is started, it will repeat continously until cancelled](#_Toc507664527)

[SaveAnalysisResultsToDatabase Property 76](#_Toc507664528)

[Whether or not to save the analysis results to the database.](#_Toc507664529)

[SaveFFTMeasurementsToDatabase Property 76](#_Toc507664530)

[True or False indicating whether to save the Fast Fourier Transform measurements to the database.](#_Toc507664531)

[SaveFinalAnalysisImages Property 77](#_Toc507664532)

[True or False indicating whether to save the final analysis images. These measurements contain overlays with potentially useful visualizations for the user.](#_Toc507664533)

[SaveFinalAnalysisImagesDelay Property 77](#_Toc507664534)

[Delay for saving the final analysis images.](#_Toc507664535)

[SaveFinalAnalysisImagesPath Property 77](#_Toc507664536)

[The location for saving the final analysis images.](#_Toc507664537)

[SaveFinalAnalysisMeasurementsToDatabase Property 77](#_Toc507664538)

[True or False indicating whether to save the final analysis measurements to the database. These measurements contain overlays with potentially useful visualizations for the user.](#_Toc507664539)

[SaveLocalContrastMeasurementsToDatabase Property 78](#_Toc507664540)

[True or False indicating whether to save the local contrast measurements to the database.](#_Toc507664541)

[SaveOnlyFailedMeasurements Property 78](#_Toc507664542)

[True or False indicating whether to save only the failed measurements to the database.](#_Toc507664543)

[SaveReferenceMeasurementsToDatabase Property 78](#_Toc507664544)

[True or False indicating whether to save the Reference measurements to the database.](#_Toc507664545)

[SaveRegisteredMeasurementsToDatabase Property 78](#_Toc507664546)

[True or False indicating whether to save the registered measurements to the database.](#_Toc507664547)

[SaveSyntheticMeasurementsToDatabase Property 78](#_Toc507664548)

[True or False indicating whether to save the synthetic measurements to the database.](#_Toc507664549)

[SerialNumberMode Property 79](#_Toc507664550)

[SerialNumberMode indicates when to enter the serial number of the device under test](#_Toc507664551)

[ShowAdvancedOptionsInPS Property 79](#_Toc507664552)

[True or False indicating whether to show advanced setup options in the Pattern Setup user interface control.](#_Toc507664553)

[ShowStatusForm Property 79](#_Toc507664554)

[True or False indicating whether to show the camera’s status form during the measurement taking process. This form shows more detailed status of the camera’s current actions.](#_Toc507664555)

[PatternBase Class 80](#_Toc507664556)

[Background 80](#_Toc507664557)

[Public Methods 80](#_Toc507664558)

[CloneMe Method 80](#_Toc507664559)

[Clones the instance of the PatternBase.](#_Toc507664560)

[ColorDescription Method 80](#_Toc507664561)

[Returns a string name for a color based on the RGB values.](#_Toc507664562)

[GenerateImage Method 80](#_Toc507664563)

[Override this function in order to use this pattern with a VideoPatternGenerator](#_Toc507664564)

[IsPatternEqualTo Method 81](#_Toc507664565)

[Uses reflection to check the properties of the pattern and see if it is equal to another pattern. Override to ignore specific properties when checking equality. The primary reason to check for equality is to tell the user whether or not they need to save the pattern object if it has been changed.](#_Toc507664566)

[Public Properties 81](#_Toc507664567)

[Name Property 81](#_Toc507664568)

[Gets or sets the pattern base name.](#_Toc507664569)

[PatternGeneratorBase Class 82](#_Toc507664570)

[Background 82](#_Toc507664571)

[Public Events 82](#_Toc507664572)

[RefreshPropertyGrid Event 82](#_Toc507664573)

[Occurs every time a request is made for the UI to update itself because PatternGeneratorBase properties were modified.](#_Toc507664574)

[Public and Protected Methods 82](#_Toc507664575)

[CloneMe Method 82](#_Toc507664576)

[Clones the instance of the PatternGeneratorBase.](#_Toc507664577)

[GetCustomForm Method 82](#_Toc507664578)

[Returns a custom form. Override this only if you also override HasCustomDeviceControlForm = True.](#_Toc507664579)

[Initialize Method 83](#_Toc507664580)

[Initializes the instance of PatternGeneratorBase](#_Toc507664581)

[IsInitializationRequired Method 83](#_Toc507664582)

[True or False whether it is required to initialize the pattern generator](#_Toc507664583)

[OnRefreshPropertyGrid Method 83](#_Toc507664584)

[Used to raise an event to the user interface requesting that it update itself, as the underlying properties of the PatternGeneratorBase have changed. Use this method if creating properties that can modify other properties.](#_Toc507664585)

[SetBrowsableAttributes Method 83](#_Toc507664586)

[Used with auto hide/show of properties in the PatternGenerator property grid. Sets initial conditions for properties to be browsable or not. Override if you want manual control over showing/hiding specific properties.](#_Toc507664587)

[SetPatternReady Method 83](#_Toc507664588)

[If your PatternGenerator requires external code to control the device and signal that the pattern is ready, have that external code call this sub to set the PatternReady flag.](#_Toc507664589)

[ShowPattern Method 84](#_Toc507664590)

[Shows a pattern. To Hide a pattern, handle the case where p is nothing.](#_Toc507664591)

[ShowRGBWPattern Method 84](#_Toc507664592)

[Shows a fullscreen red, green, blue, or white color pattern. This Sub is only called from the Four Color Calibration form. If your device does not have RGB primaries, you do not need to override this Sub.](#_Toc507664593)

[ShutDown Method 84](#_Toc507664594)

[Shuts down the pattern generator.](#_Toc507664595)

[Public Properties 84](#_Toc507664596)

[DelayMilliseconds Property 84](#_Toc507664597)

[Gets or sets the number of milleseconds to wait for the pattern to be rendered on the display.](#_Toc507664598)

[HasCustomDeviceForm Property 85](#_Toc507664599)

[Override and set this property to True if you want to develop a custom form to control the PatternGenerator. The custom form will appear in the View menu of TrueTest.](#_Toc507664600)

[IsInitialized Property 85](#_Toc507664601)

[This property is set to True after the Initialize method is called. It can be overridden for tighter control over the initialization process.](#_Toc507664602)

[SolidColorPattern Class 86](#_Toc507664603)

[Background 86](#_Toc507664604)

[Constructors 86](#_Toc507664605)

[New 86](#_Toc507664606)

[Initializes a new instance of the SolidColor Class.](#_Toc507664607)

[Public Properties 86](#_Toc507664608)

[PatternColor Property 86](#_Toc507664609)

[Gets or sets the pattern color.](#_Toc507664610)

[Common Enumerations – TrueTestEngine Namespace 87](#_Toc507664611)

[TrueTest.APIResult Enum 87](#_Toc507664612)

[TrueTest.SequenceModeEnum Enum 87](#_Toc507664613)

[TrueTest.AnalysisResultEnum Enum 87](#_Toc507664614)

[TrueTest.OperatingModeEnum Enum 87](#_Toc507664615)

[TrueTest.AnalysisImageType Enum 87](#_Toc507664616)

[TrueTest.CameraRotation Enum 87](#_Toc507664617)

[Common Enumerations – RadiantCommon Namespace 88](#_Toc507664618)

[RiBitmapCtl.BlobDrawShape Enum 88](#_Toc507664619)

[ColorSpace Enum 88](#_Toc507664620)

[DistanceScaleTypeEnum 88](#_Toc507664621)

[DistanceUnitTypeEnum 88](#_Toc507664622)

[PhotometricTermType Enum 88](#_Toc507664623)

[PhotometricUnitTypeEnum 88](#_Toc507664624)

[SpectralResponseTypeEnum 89](#_Toc507664625)

# INTRODUCTION

This document describes the API for TrueTestEngine and TrueTestPatternGenerator. TrueTestEngine and TrueTestPatternGenerator are compatible with .NET Framework 4.0. Some classes in this document may refer to classes from other DLLs and Namespaces, such as PMEngine and RadiantCommon. For documentation on these namespaces, refer to the PMEngine API.

The basic steps to incorporate the TrueTestEngine API into your program are given below. Pick the steps based on what type of project you are creating: either a new Analysis, Pattern, or Pattern Generator, or a new User Interface:

## Initial Setup (Analysis):

1. Create your project or copy and rename an existing example from the API package.
2. Go to your project references and confirm the reference to RadiantCommon.dll, TrueTestEngine.dll, and Radiant.AssemblyLoader.dll are there, otherwise add them (Use the browse method to browse the dlls in TrueTest installation directory (C:\Program Files\Radiant Vision Systems\TrueTest 1.7, or earlier version.)
3. Select to include the Imported namespaces: RadiantCommon, TrueTestEngine, and TrueTestPatternGenerator.
4. Create a class, and inherit from one of three inheritable base classes:

TrueTestEngine.AnalysisBase

1. Fill in the MustOverride functions of the base class.

## Initial Setup (Pattern, Pattern Generator):

1. Create your project or copy and rename an existing example from the API package.
2. Go to your project references and confirm the reference to RadiantCommon.dll, TrueTestEngine.dll, TrueTestPatternGenerator.dll, and Radiant.AssemblyLoader.dll are there, otherwise add them (Use the browse method to browse the dlls in TrueTest installation directory (C:\Program Files\Radiant Vision Systems\TrueTest 1.7, or earlier version).)
3. Select to include the Imported namespaces: RadiantCommon, TrueTestEngine, and TrueTestPatternGenerator.
4. Create a class, and inherit from one of three inheritable base classes:

TrueTestPatternGenerator.PatternBase

TrueTestPatternGenerator.PatternGeneratorBase

1. Fill in the MustOverride functions of the base class.

## Initial Setup (User Interface):

1. Create your project.
2. Go to your project references and confirm/add a reference to the DLLs:TrueTestEngine.dll, TrueTestPatternGenerator.dll, RadiantCommon.dll, PMEngine.dll.
3. Import namespaces for RadiantCommon, PMEngine, TrueTestEngine, and TrueTestPatternGenerator.

##### Example

For an example User Interface in VB.NET, please see the example project included with this document, TrueTestEngine\_API\_Example.vbproj. Follow the Initial Setup above for a User Interface.

# TrueTest Class

## Background

*TrueTestEngine Namespace*

The TrueTest class is the heart of the TrueTest API. It contains mostly shared properties and methods, so there is no need to carry a local copy of the TrueTest class.

Access to global Settings, control of running/stopping the Sequence, and event handling can all be found in the TrueTest shared class.

## Public Methods

### AppSettings Method

#### Retrieves the current set of global Settings.

##### Syntax

###### Declaration

Public Shared Function AppSettings() As TrueTestEngine.Settings

###### Returns

A Settings class containing the current global Settings.

###### Remarks

### CameraCalibrationDatabaseSetWithBrowse Method

#### Lets the user browse to set a \*.calx camera calibration database file with the desired filename. The filename the user selected is returned via the argument CalibrationDBpath. This requires a restart of the software, as changing the calibration database also implies changing the camera.

##### Syntax

###### Declaration

Public Shared Function CameraCalibrationDatabaseSetWithBrowse(ByRef *CalibrationDBpath* As String, *NewDB* as Boolean) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*CalibrationDBpath*

The filename and path for the \*.calx file to be set

*NewDB*

When set to True, the software restarts after the calibration database is changed, which is required as changing calibration databases implies changing cameras.

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

An application restart is required to connect to a different camera.

### ClearMeasurementHistory Method

#### Clears the measurement history saved in the current measurement database. This deletes all generated measurements (Local Contrast, RADA, Reference, FFT, etc), and deletes any Results. In effect, it cleans the database of all data except original camera measurements. Optionally you can clear the measurement history for an individual serial number.

##### Syntax

###### Declaration

Public Shared Sub ClearMeasurementHistory(Optional *SerialNumber* As String = “”)

###### Parameters

*SerialNumber*

(Optional) The serial number to clear from the measurement database. If set to the empty string, all serial numbers will be cleared.

###### Remarks

### Initialize Method

#### Initializes all components of TrueTest. Call this during the load process of the application. Requires input of three database file paths to initialize.

##### Syntax

###### Declaration

Public Shared Function Initialize(ByRef *CalibrationDBList* As List(of String), ByRef *SequenceDBpath* As String, ByRef *MeasurementDBpath* As String, ByRef *MDIForm* as System.Windows.Forms.Form) As Integer

###### Parameters

*CalibrationDBList*

The file path for the camera calibration database. If the file doesn’t exist, the user will be asked to select a new file, and this parameter will contain the user’s selection.

*SequenceDBpath*

The file path for the \*.seqx sequence file. If the file doesn’t exist, the user will be asked to select a new file, and this parameter will contain the user’s selection.

*MeasurementDBpath*

The file path for the \*.ttxm measurement database. If the file doesn’t exist, the user will be asked to select a new file, and this parameter will contain the user’s selection.

*MDIForm*

The main user interface form to be used with TrueTest. In VB syntax, if you are initializing TrueTest from the main form’s load event, use the keyword ‘Me’ (or ‘this’ in C#).

###### Returns

A Boolean indicating whether or not the function was successful.

###### Remarks

This is the preferred method to initialize TrueTest, and is also the cleanest.

### InitializeAppSettings Method

#### Resets the global Settings class to the most recent Settings that were saved.

##### Syntax

###### Declaration

Public Shared Function InitializeAppSettings() As Boolean

###### Returns

A Boolean indicating whether or not the function was successful.

###### Remarks

This method only initializes the Settings (found in the user interface under Tools/Options). The preferred method of initializing TrueTest is to call TrueTest.Initialize.

### InitializeCamera Method

#### Connects to and initializes a camera using the desired camera calibration database.

##### Syntax

###### Declaration

Public Shared Function InitializeCamera(*CameraCalibrationFilePathName* As String) As Integer

###### Parameters

*CameraCalibrationFilePathName*

The file path pointing to the camera calibration database.

###### Returns

An integer indicating whether or not the function was successful.

0 = Successful

-1 = Failed; could not initialize camera (e.g. not powered)

-2 = Security check did not pass (e.g. calibration database serial number did not match camera serial number)

-3 = Camera overheated; power cycle required

-4 = Camera firmware updated; power cycle required

###### Remarks

This method only initializes the camera. The preferred method of initializing TrueTest is to call TrueTest.Initialize.

### MeasurementDatabaseNew Method

#### Creates a new \*.ttxm measurement database file with the desired filename.

##### Syntax

###### Declaration

Public Shared Function MeasurementDatabaseNew(*MeasurementDatabaseFilePathName*

As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*MeasurementDatabaseFilePathName*

The filename and path for the new \*.ttxm file to be created

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### MeasurementDatabaseNewWithBrowse Method

#### Lets the user browse to creates a new \*.ttxm measurement database file with the desired filename. The filename is returned via the argument MeasurementDBpath.

##### Syntax

###### Declaration

Public Shared Function MeasurementDatabaseNewWithBrowse(ByRef *MeasurementDBpath* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*MeasurementDBpath*

The filename and path for the new \*.ttxm file to be created

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### MeasurementDatabaseSet Method

#### Sets the \*.ttxm measurement database file to the desired filename.

##### Syntax

###### Declaration

Public Shared Function MeasurementDatabaseSet(*MeasurementDatabaseFilePathName*

As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*MeasurementDatabaseFilePathName*

The filename and path for the \*.ttxm file to be set

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### MeasurementDatabaseSetWithBrowse Method

#### Lets the user browse to set a \*.ttxm measurement database file with the desired filename. The filename the user selected is returned via the argument MeasurementDBpath.

##### Syntax

###### Declaration

Public Shared Function MeasurementDatabaseSetWithBrowse(ByRef *MeasurementDBpath* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*MeasurementDBpath*

The filename and path for the \*.ttxm file to be set

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### MeasurementDatabaseSetWithLoop Method

#### Lets the user browse to set a \*.ttxm measurement database file with the desired filename. The filename the user selected is returned via the argument MeasurementDBpath. If the initial argument for MeasurementDBpath cannot be found, the user is given a choice of browsing to select a database or browsing to select a path to create a new database.

##### Syntax

###### Declaration

Public Shared Function MeasurementDatabaseSetWithLoop(ByRef *MeasurementDBpath* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*MeasurementDBpath*

The filename and path for the \*.ttxm file to be set

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### SequenceNew Method

#### Creates a new \*.seqx Sequence file with the desired filename.

##### Syntax

###### Declaration

Public Shared Function SequenceNew(*SequenceFilePathName* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SequenceFilePathName*

The filename and path for the new \*.seqx file to be created

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### SequenceNewWithBrowse Method

#### Lets the user browse to select a new \*.seqx Sequence file, starting in the folder containing the desired filename. The filename is returned By Reference in the argument SequenceDBpath.

##### Syntax

###### Declaration

Public Shared Function SequenceNewWithBrowse(ByRef *SequenceDBpath* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SequenceDBpath*

The filename and path for the new \*.seqx file to be created

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### SequenceRunAll Method

#### Runs all steps in the current sequence from the beginning. Default run mode is Synchronous.

##### Syntax

###### Declaration

Public Shared Sub SequenceRunAll(Optional *Mode* As TrueTestEngine.TrueTest.SequenceModeEnum)

###### Parameters

*Mode*

(Optional) The mode – synchronous or asynchronous – to use when running the sequence.

###### Remarks

This method also clears the ObjectRepository before running any steps; the other Run methods do not clear the Object Repository.

### SequenceSave Method

#### Saves the current sequence to the file path SetupXMLFilePathName. Equivalent to File->Save As sequence. Also equivalent to File->Save if the file path name already exists on disk (overwrites).

##### Syntax

###### Declaration

Public Shared Function SequenceSave(*SetupXMLFilePathName* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SetupXMLFilePathName*

The path to the \*.seqx file where the sequence should be saved.

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### SequenceSaveRequired Method

#### Returns whether or not the sequence has been edited since its last save.

##### Syntax

###### Declaration

Public Shared Function SequenceSaveRequired() As Boolean

###### Returns

Whether or not the sequence has been edited.

###### Remarks

### SequenceSet Method

#### Sets the current sequence using the \*.seqx file found at SetupXMLFilePathName. Equivalent to File->Open sequence.

##### Syntax

###### Declaration

Public Shared Function SequenceSet(*SetupXMLFilePathName* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SetupXMLFilePathName*

The path to the \*.seqx file containing the sequence to open.

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### SequenceSetWithBrowse Method

#### Sets the current sequence using the \*.seqx file found at SequenceDBpath. Equivalent to File->Open sequence. If no file is found at SequenceDBpath, then the user is presented with a browse dialog.

##### Syntax

###### Declaration

Public Shared Function SequenceSetWithBrowse(ByRef *SequenceDBpath* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SequenceDBpath*

The path to the \*.seqx file containing the sequence to open.

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

If the user browses to find a file path, it is saved by reference into SequenceDBpath

### SequenceSetWithLoop Method

#### Sets the current sequence using the \*.seqx file found at SequenceDBpath. Equivalent to File->Open sequence. If no file is found at SequenceDBpath, then the user is presented with a browse dialog. If the user tries to cancel the dialog, they are asked if they want to try again or browse to create a new sequence.

##### Syntax

###### Declaration

Public Shared Function SequenceSetWithLoop(ByRef *SequenceDBpath* As String, Optional *customMessage* As String) As TrueTestEngine.TrueTest.APIResultEnum

###### Parameters

*SequenceDBpath*

The path to the \*.seqx file containing the sequence to open.

*customMessage*

If no custom message is entered, the default “’Couldn’t find sequence:’ + *SequenceDBPath*” will be used

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

If the user browses to find a file path or create a new sequence, it is saved by reference into SequenceDBpath.

### SequenceStop Method

#### Stops the current sequence. If the user wants to cancel, call this method.

##### Syntax

###### Declaration

Public Shared Sub SequenceStop()

###### Remarks

### SetPatternReady Method

#### Calls the ‘SetPatternReady’ method of the Pattern generator. Used if the pattern generator is designed to wait for this response before continuing.

##### Syntax

###### Declaration

Public Shared Sub SetPatternReady()

###### Remarks

### ShowEditSequenceDialog Method

#### Shows a dialog allowing the user to edit the sequence. Use only if you are creating a custom UI and want to easily control the ability to edit a sequence.

##### Syntax

###### Declaration

Public Shared Sub ShowEditSequenceDialog()

###### Remarks

A dialog will appear allowing the user to edit and save the Sequence.

### ShowPatternGeneratorControlForm Method

#### Shows a dialog allowing the user to edit pattern generator settings. Only shows a dialog if the PatternGenerator property HasCustomDeviceControlForm is set to True, and the method GetCustomForm was overridden properly.

##### Syntax

###### Declaration

Public Shared Sub ShowPatternGeneratorControlForm()

###### Remarks

If created by the programmer, a dialog will appear allowing the user to edit PatternGenerator properties.

### ShowSerialNumberDialog Method

#### Shows a dialog that allows the user to enter a serial number. Includes some optional parameters for showing and saving the serial number.

##### Syntax

###### Declaration

Public Shared Function ShowSerialNumberDialog(Optional ByRef *SerialNumber* As String = “”, Optional *RequireSerialNumber* As Boolean = True, Optional *AllowDuplicateSerialNumbers* as Boolean = False) As System.Windows.Forms.DialogResult

###### Parameters

*SerialNumber*

(ByRef) The initial serial number to show the user. Also returns the serial number that the user entered. Note that the serial number can always be retrieved from the property TrueTest.SerialNumber.

*RequireSerialNumber*

Whether or not to require the user to enter a serial number, or allow the empty string as a possible serial number. If the empty string is used, TrueTest will replace it with a timestamp of format “yyyyMMddHHmmss”.

*AllowDuplicateSerialNumbers*

Whether or not to allow the user to enter the same serial number twice.

###### Returns

The DialogResult from the form, indicating whether the user clicked OK, Cancel, etc.

###### Remarks

### Shutdown Method

#### Shutsdown the camera and clears TrueTest settings.

##### Syntax

###### Declaration

Public Shared Sub Shutdown()

###### Remarks

Should only be called when closing the application. To use TrueTest again, it will need to be Initialized.

### TakeMeasurement Method

#### Shows a dialog allowing the user to select a PatternSetup and take a manual measurement. The measurement will be saved as an original camera measurement into the measurement database, but will not be sent to an Analysis.

##### Syntax

###### Declaration

Public Shared Function TakeMeasurement() As TrueTestEngine.TrueTest.APIResultEnum

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

### TakeMeasurement Method

#### Takes a measurement using the PatternSetup provided. The measurement will be returned and also saved as an original camera measurement, but will not be sent to an Analysis.

##### Syntax

###### Declaration

Public Shared Function TakeMeasurement(*ps* As TrueTestEngine.PatternSetup, *SerialNbr* As String, *SequenceIndex* As Integer, *UsePreviousMeasurement* As Boolean, Optional *SecondSequenceIndex* as Integer = -1) As PMEngine.PMMeasurement

###### Parameters

*ps*

The PatternSetup to use for taking the measurement.

*SerialNbr*

The serial number to associate this measurement with.

*SequenceIndex*

The sequence number to associate this measurement with.

*UsePreviousMeasurement*

Whether or not to look for another measurement in the measurement buffer with a matching PatternSetup name and serial number. If this parameter is True and such a measurement is found, it will be retrieved instead of the camera taking a new measurement.

*SecondSequenceIndex*

(Optional) A second sequence index to associate this measurement with. Not necessary to set when using TakeMeasurement in this context.

###### Returns

Whether or not the operation was successful, failed, or canceled.

###### Remarks

Parameters such as SerialNbr and SequenceIndex are not critical to how the measurement is taken in this context, but can be useful metadata for a measurement when a user tries to use TrueTest in Database (Auto) mode.

### UIForm Method

#### Returns the user interface form connected to TrueTest.

##### Syntax

###### Declaration

Public Shared Sub UIForm() as System.Windows.Forms.Form

###### Returns

The main user interface Form.

###### Remarks

## Public Properties

### CalibrationFilePathName Property (ReadOnly)

#### Gives access to the currently selected Calibration database (\*.calx) where camera calibration information is stored.

##### Syntax

###### Declaration

Public Shared ReadOnly Property CalibrationFilePathName() As String

###### Property Value

A string corresponding to the file path for the calibration database.

###### Remarks

### MeasurementDatabasePathFileName Property (ReadOnly)

#### Gives access to the currently selected Measurement database (\*.ttxm) where measurements and results are stored.

##### Syntax

###### Declaration

Public Shared ReadOnly Property MeasurementDatabasePathFileName() As String

###### Property Value

A string corresponding to the file path for the measurement database.

###### Remarks

### OperatingMode Property

#### Sets the current mode of operation: Use Camera, Database (Auto), Database (User Select). Database modes are most often used for troubleshooting or testing.

##### Syntax

###### Declaration

Public Shared Property OperatingMode() As TrueTestEngine.TrueTest.OperatingModeEnum

###### Property Value

The currently chosen Operating Mode.

###### Remarks

### PatternGenerator Property

#### Gives access to the current Pattern Generator object used to generate patterns to be measured on the device under test. The Pattern Generator can also include any external hardware control commands.

##### Syntax

###### Declaration

Public Shared Property PatternGenerator() As TrueTestPatternGenerator.PatternGeneratorBaseWrapper

###### Property Value

The PatternGeneratorBaseWrapper containing the Pattern Generator.

###### Remarks

The Pattern Generator is wrapped for internal purposes. To access the Pattern Generator, which inherits from TrueTestPatternGenerator.PatternGeneratorBase, use the syntax TrueTest.PatternGenerator.PatternGenerator.

### PMEngine Property

#### Gives access to PMEngine for direct control of the camera.

##### Syntax

###### Declaration

Public Shared Property PMEngine() As PMEngine.PMEngine

###### Property Value

The PMEngine object for camera control.

###### Remarks

Most actions can be done using the TrueTest API, including taking measurements. For deeper control, the PMEngine API can be used. Refer to the PMEngine .NET API (separately documented) for more information about PMEngine.

### SaveMeasurementsToDatabase Property

#### Sets whether or not to save camera measurements to the database.

##### Syntax

###### Declaration

Public Shared Property SaveMeasurementsToDatabase() As Boolean

###### Property Value

Whether or not to save measurements to the database.

###### Remarks

Only applies to original camera measurements. Some processing of the measurements may result in other Measurement objects that may still be written to the database. For options on whether or not to save these measurements, see the Settings class.

### Sequence Property

#### Gives access to the current Sequence, which contains all the setup information for taking and analyzing measurements. Each Sequence is saved in a separate sequence file with the \*.seqx extension.

##### Syntax

###### Declaration

Public Shared ReadOnly Property Sequence(Optional *ChannelIndex* As Integer = -1) As Sequence

###### Property Value

The current Sequence, containing all the test setup information.

###### Parameters

*ChannelIndex*

Optional integer to identify the channel number of the sequence being requested. The default value is “-1”, returning the active sequence.

###### Remarks

### SerialNumber Property

#### Gives access to the most recently entered Serial Number of a device under test.

##### Syntax

###### Declaration

Public Shared Property SerialNumber(Optional *ChannelIndex* As Integer = -1) As String

###### Property Value

A string corresponding to the most recently tested serial number.

###### Parameters

*ChannelIndex*

Optional integer to identify the channel number of the sequence being requested. The default value is “-1”, returning the active sequence.

###### Remarks

### UserDataFolder Property

#### Gets the User Data Folder.

##### Syntax

###### Declaration

Public Shared ReadOnly Property UserDataFolder() As String

###### Property Value

A string corresponding to the User Data Folder path.

###### Remarks

## Public Events

### \_AnalysisComplete Event

#### Occurs when an analysis has completed.

##### Syntax

###### Declaration

Public Shared Event \_AnalysisComplete(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) that has completed.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_DefectListChanged Event

#### Occurs when an analysis is requesting to show a defect list to the user.

##### Syntax

###### Declaration

Public Shared Event \_DefectListChanged(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) requesting to show the defect list to the user.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_MeasurementChanged Event

#### Occurs when a measurement has had additional processing, and is requested to be shown to the user.

##### Syntax

###### Declaration

Public Shared Event \_MeasurementChanged(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence), or the current Analysis object (of type AnalysisBase)

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

Handle this event if you are creating a custom User Interface and wish to show all processing results to the user.

### \_MeasurementComplete Event

#### Occurs when a camera measurement has been fully completed and is about to be passed to the Analysis.

##### Syntax

###### Declaration

Public Shared Event \_MeasurementComplete(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence)

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_ROIArrayChanged Event

#### Occurs when an anlaysis is requesting to show an ROI array to the user.

##### Syntax

###### Declaration

Public Shared Event \_ROIArrayChanged(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) requesting to show the ROI array to the user.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_ShowMeasurement Event

#### Occurs when an analysis is requesting to show a measurement to the user.

##### Syntax

###### Declaration

Public Shared Event \_ShowMeasurement(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) requesting to show the measurement.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_ShowCustomForm Event

#### Occurs when an analysis wishes to show a Custom form (different from the standard bitmap form) to the user.

##### Syntax

###### Declaration

Public Shared Event \_ShowCustomForm(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) that wishes to show the custom form.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### \_UpdateCustomForm Event

#### Occurs when an analysis wishes to update a Custom form (different from the standard bitmap form) already shown to the user.

##### Syntax

###### Declaration

Public Shared Event \_UpdateCustomForm(*sender* As Object, *e* As String)

###### Parameters

*sender*

The Analysis object (TrueTest.AnalysisBase) that wishes to update the custom form.

*e*

The string Key used to get arguments from the ObjectRepository. See EventArgs for the corresponding class type that the object in the repository will need to be converted into.

###### Remarks

### ExposureComplete Event

#### Occurs every time an exposure within a camera measurement has been completed.

##### Syntax

###### Declaration

Public Shared Event \_MeasurementComplete(*sender* As Object, *e* As TrueTestEngine.ExposureCompleteEventArgs)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence)

*e*

The EventArgs class containing information about an exposure that has completed. See EventArgs in this document for more information on the EventArgs classes.

###### Remarks

If used properly, external hardware can be triggered to move a device under test as soon as the camera has finished its exposure, instead of waiting for \_AnalysisComplete or \_SequenceComplete, to improve efficiency.

### MeasurementDatabaseChanged Event

#### Occurs when the user has selected a new measurement database file (\*.ttxm).

##### Syntax

###### Declaration

Public Shared Event MeasurementDatabaseChanged(*sender* As Object, *filepath* As String)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence)

*filepath*

The full file path of the measurement database file that has been loaded by TrueTest.

###### Remarks

### SequenceChanged Event

#### Occurs when the user has selected a new sequence file (\*.seqx).

##### Syntax

###### Declaration

Public Shared Event SequenceChanged(*sender* As Object, *filepath* As String)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence)

*filepath*

The full file path of the sequence file that has been loaded by TrueTest.

###### Remarks

### SequenceComplete Event

#### Occurs when a sequence has completed.

##### Syntax

###### Declaration

Public Shared Event SequenceComplete(*sender* As Object, *e* As TrueTestEngine.SequenceCompleteEventsArgs)

###### Parameters

*sender*

The Sequence object (TrueTest.Sequence)

*e*

The EventArgs class containing information about a sequence that has completed. See EventArgs in this document for more information on the EventArgs classes.

###### Remarks

### ShowPatternCommandSent Event

#### Occurs when the Pattern Generator is trying to Show a Pattern.

##### Syntax

###### Declaration

Public Shared Event ShowPatternCommandSent(*sender* As Object, *e* As TrueTestPatternGenerator.ShowPatternCommandSentEventArgs)

###### Parameters

*sender*

The object sending the event.

*e*

The EventArgs corresponding to this event. These contain the Pattern object to be shown.

###### Remarks

The main reason to handle this event is if you are building a custom Pattern Generator or interface to external hardware and do not have direct control over the device. If you handle this event, you should call SetPatternReady to signal to the Pattern Generator that the pattern has been set, and is ready for measurement.

# AnalysisBase Class

## Background

*TrueTestEngine Namespace*

The MustInherit AnalysisBase Class is the starting point for developing custom analyses. The Execute method will be called by TrueTestEngine to execute the desired analysis. You can add custom properties for users to set to shape how the analysis is performed. Any public properties added to a class inheriting from AnalysisBase must have default values, as they will eventually be saved into the TrueTest sequence file (\*.seqx).

## MustOverride Methods and Properties

### Execute Method

#### Executes the code that performs the analysis. Override this function and put image processing code here.

##### Syntax

###### Declaration

Protected Overrides Sub Execute()

###### Remarks

### Name Property (ReadOnly)

#### Sets the text Name that will be written to the Results database for results of the Analysis.

##### Syntax

###### Declaration

Public Overrides ReadOnly Property Name() As String

###### Property Value

Name of the Analysis type.

###### Remarks

This name should be unique compared to all other Analyses.

### UserName Property

#### Sets the initial User Name that the user gets to edit.

##### Syntax

###### Declaration

Public Overrides Property UserName() As String

###### Property Value

User editable name for a specific instance of this analysis in a sequence.

###### Remarks

## Protected Properties

### SequenceIndex Property

#### The current index of the analysis within the Sequence class list Items. Can be used to know which step in the sequence is currently executing.

##### Syntax

###### Declaration

Protected Friend Property SequenceIndex() As Integer

###### Property Value

The index of the analysis in the Sequence.

###### Remarks

### SecondSequenceIndex Property

#### A secondary index. Used if an Analysis wishes to maintain multiple bitmap windows.

##### Syntax

###### Declaration

Protected Property SecondSequenceIndex() As Integer

###### Property Value

The second index of the analysis in the Sequence.

###### Remarks

### PassFail Property

#### Shows whether or not the analysis has passed or failed.

##### Syntax

###### Declaration

Protected Friend Property PassFail() As TrueTest.AnalysisResultEnum

###### Property Value

Whether or not the test has passed or failed. Can also be set to “None” to denote neither pass nor fail.

###### Remarks

This property is not required to be set within an Analysis’ Execute sub. The ResultList will be automatically checked for any Results that passed or failed, and this information will be used to set the PassFail property.

### PatternNameList Property

#### Contains a list of strings corresponding to the names of the patterns used to generate the measurements being passed in to this Analysis.

##### Syntax

###### Declaration

Protected Friend Property PatternNameList() As List(Of String)

###### Property Value

A list of strings corresponding to pattern names used for this analysis.

###### Remarks

### PatternMeasurementList Property

#### Contains a list of measurements for this Analysis to execute with. The list type is of class PatternMeasurement.

##### Syntax

###### Declaration

Protected Friend Property PatternMeasurementList() As List(Of TrueTestEngine.PatternMeasurement)

###### Property Value

A list of strings corresponding to pattern names used for this analysis.

###### Remarks

One of the first actions inside an Analysis’ Execute function should be to pull out a MeasurementF object from this list, e.g. Dim m as MeasurementF = PatternMeasurementList(0).CurrentMeasurement

### ResultList Property

#### Contains a list of Result objects to be written to the database and potentially used to generate a report.

##### Syntax

###### Declaration

Protected Friend Property ResultList() As List(Of TrueTestEngine.Result)

###### Property Value

A list of result information generated by the analysis.

###### Remarks

If there is a result to be reported for this analysis, it should be added to the list. For instance: ResultList.Add(New Result(… ….))

### ROIArray Property

#### Contains a list of ROI objects.

##### Syntax

###### Declaration

Protected Friend Property ROIArray() As RegionOfInterest()

###### Property Value

The ROI array.

###### Remarks

## Protected Methods

### Deserialize Method

#### Used to deserialize objects from XML strings when saving custom properties of an Analysis.

##### Syntax

###### Declaration

Protected Function Deserialize(*stringValue* as string, *type* as Type) As Object

###### Parameters

*stringValue*

The XML formatted string to deserialize.

*type*

The Type for the object to deserialize the string into.

###### Returns

An object with its public properties set to the information contained in the stringValue.

###### Remarks

Currently this is only used for type System.Drawing.Color.

### OnMeasurementChanged Method

#### Used to raise MeasurementChanged event from within an Execute function.

##### Syntax

###### Declaration

Protected Sub OnMeasurementChanged(*e* As MeasurementChangedEventArgs)

###### Parameters

*e*

The EventArgs class containing information about an exposure that has completed. See EventArgs in this document for more information on the EventArgs classes.

###### Returns

Raises the MeasurementChanged event.

###### Remarks

### OnROIArrayChanged Method

#### Used to raise ROIArrayChanged event from within an Execute function.

##### Syntax

###### Declaration

Protected Sub OnROIArrayChanged()

###### Returns

Raises the ROIArrayChanged event.

###### Remarks

### Serialize Method

#### Used to serialize objects to XML strings when saving custom properties of an Analysis.

##### Syntax

###### Declaration

Protected Function Serialize(obj as Object) As String

###### Parameters

*obj*

The object to serialize.

###### Returns

An XML formatted string containing the Public information of obj.

###### Remarks

Currently this is only used for type System.Drawing.Color.

### SetAnalysisLevelPassFail Method

#### Sets the PassFail property of the AnalysisBase based on the pass/fail values inside the AnalysisBase property ResultList. Can be overridden if the programmer wishes to ignore certain results, or set the PassFail property manually. This method is called automatically by TrueTestEngine during runtime.

##### Syntax

###### Declaration

Protected Overrides Sub SetAnalysisLevelPassFail()

###### Remarks

### WritePMMeasurementToDatabase Method

#### Writes a RadiantCommon.MeasurementF object to the measurement database (\*.ttxm file) for further manual analysis.

##### Syntax

###### Declaration

Protected Sub WritePMMeasurementToDatabase(*mf* As MeasurementF, *Description* As String)

###### Parameters

*mf*

The measurement to be written to the database.

*Description*

The description of the measurement to be written to the database.

###### Remarks

Can be used inside the Execute sub to write intermediate results to the database, such as what the measurement looks like after only partial processing.

# EventArgs Classes

## Background

*TrueTestEngine Namespace*

The EventArgs series of classes are containers used to pass objects from TrueTestEngine up to the top-level user interface through events. Each of the EventArgs classes should be aptly named to understand which event they correspond to.

Most events come from the class TrueTest. Refer to this class to find when the events are fired.

## AnalysisCompleteEventArgs

### AnalysisUserName Property

#### Contains the text name of the Analysis that was completed.

##### Syntax

###### Declaration

Public Property AnalysisUserName() As String

###### Property Value

Name of the Analysis performed.

###### Remarks

### PassFail Property

#### Reports whether the Analysis passed or failed.

##### Syntax

###### Declaration

Public Property PassFail() As TrueTestEngine.TrueTest.AnalysisResultEnum

###### Property Value

The analysis may Pass, Fail, or have a result of None (no pass/fail criteria were set).

###### Remarks

### PatternName Property

#### Reports the pattern name.

##### Syntax

###### Declaration

Public Property PatternName() As String

###### Property Value

The analysis pattern name.

###### Remarks

### Results Property

#### Contains a list of Results from the Analysis that has completed.

##### Syntax

###### Declaration

Public Property Results() As List(Of TrueTestEngine.Result)

###### Property Value

A list of Results from the analysis.

###### Remarks

See the Result class in this document for more information

### SequenceIndex Property

#### The index number of the analysis which has completed.

##### Syntax

###### Declaration

Public Property SequenceIndex() As Integer

###### Property Value

The index number of the analysis that was completed.

###### Remarks

## DefectListChangedEventArgs

### AppendToList Property

#### Reports whether or not the Defect List should be appended to an existing list.

##### Syntax

###### Declaration

Public Property AppendToList() As Boolean

###### Property Value

Whether or not to append the Defect List to an existing list.

###### Remarks

This name should be unique compared to all other Analyses.

### BlobDrawShape Property

#### Reports the shape to draw if blobs are found inside the Defect List.

##### Syntax

###### Declaration

Public Property BlobDrawShape() As RadiantCommon.RiBitmapCtl.BlobDrawShapeEnum

###### Property Value

The type of shape to draw for any blobs inside the Defect List.

###### Remarks

### DefectList Property

#### The list of Defects. The purpose of this list is to get drawn on a RadiantCommon.RiBitmapCtl to show the user where defects were found.

##### Syntax

###### Declaration

Public Property DefectList() As List(Of RadiantCommon.DefectBase)

###### Property Value

A list of defect information.

###### Remarks

### Measurement Property

#### A camera measurement object related to the defects in DefectList.

##### Syntax

###### Declaration

Public Property Measurement() As RadiantCommon.MeasurementBase

###### Property Value

A measurement object related to the DefectList.

###### Remarks

### SequenceIndex Property

#### Reports the index of the analysis in the sequence that the DefectList came from.

##### Syntax

###### Declaration

Public Property SequenceIndex() As Integer

###### Property Value

The index of the analysis where the DefectList came from.

###### Remarks

### SecondSequenceIndex Property

#### Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.

##### Syntax

###### Declaration

Public Property SecondSequenceIndex() As Integer

###### Property Value

The index of the user interface control where the DefectList should be updated.

###### Remarks

## ExposureCompleteEventArgs

### SequenceIndex Property

#### Reports the index of the analysis in the sequence that the DefectList came from.

##### Syntax

###### Declaration

Public Property SequenceIndex() As Integer

###### Property Value

The index of the analysis where the DefectList came from.

###### Remarks

### NextPattern Property

#### Contains the PatternBase object that the sequence wants to be shown before taking the next camera measurement.

##### Syntax

###### Declaration

Public Property NextPattern() As TrueTestPatternGenerator.PatternBase

###### Property Value

The next pattern to be shown before taking another measurement.

###### Remarks

## MeasurementCompleteEventArgs

### Measurement Property

#### A camera measurement object returned when the camera has completed taking a measurement.

##### Syntax

###### Declaration

Public Property Measurement() As RadiantCommon.MeasurementBase

###### Property Value

A measurement object.

###### Remarks

### PatternSetup Property

#### The collection of properties used to take the measurement which has completed.

##### Syntax

###### Declaration

Public Property PatternSetup() As TrueTestEngine.PatternSetup

###### Property Value

The setup properties used to take the completed measurement.

###### Remarks

### SequenceIndex Property

#### Reports the index of the analysis in the sequence that the Measurement came from.

##### Syntax

###### Declaration

Public Property SequenceIndex() As Integer

###### Property Value

The index of the analysis where the completed Measurement came from.

###### Remarks

### SecondSequenceIndex Property

#### Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.

##### Syntax

###### Declaration

Public Property SecondSequenceIndex() As Integer

###### Property Value

The index of the user interface control where the Measurement should be updated.

###### Remarks

## ROIArrayChangedEventArgs

### ROIArray Property

#### An array of Regions Of Interest (ROIs) to be displayed to the user. ROIs can be circles, rectangles, ellipses, or polygons.

##### Syntax

###### Declaration

Public Property ROIArray() As RadiantCommon.RegionOfInterest()

###### Property Value

The array of RegionOfInterest objects to be passed along to the user interface.

###### Remarks

### SequenceIndex Property

#### Reports the index of the analysis in the sequence that the ROIArray came from.

##### Syntax

###### Declaration

Public Property SequenceIndex() As Integer

###### Property Value

The index of the analysis where the ROIArray came from.

###### Remarks

### SecondSequenceIndex Property

#### Reports a second index in case an analysis has opened more than one form containing a RadiantCommon.RiBitmapCtl. Can be used to distinguish between two different user interface controls to update for the same analysis.

##### Syntax

###### Declaration

Public Property SecondSequenceIndex() As Integer

###### Property Value

The index of the user interface control where the ROIArray should be updated.

###### Remarks

## SequenceCompleteEventsArgs

### Canceled Property

#### Reports if the sequence has been canceled.

##### Syntax

###### Declaration

Public Property Canceled() As Boolean

###### Property Value

True or False, whether the sequence has been canceled.

###### Remarks

### NumFailedTests Property

#### Reports the number of tests in the completed sequence that have failed.

##### Syntax

###### Declaration

Public Property NumFailedTests() As Integer

###### Property Value

Number of tests that failed in the sequence that completed.

###### Remarks

### PassFail Property

#### Reports whether or not the entire sequence passed or failed.

##### Syntax

###### Declaration

Public Property PassFail() As TrueTestEngine.TrueTest.AnalysisResultEnum

###### Property Value

Whether or not the sequence passed, failed, or no pass/fail criteria were set.

###### Remarks

### SequenceName Property

#### Reports the name of the completed sequence.

##### Syntax

###### Declaration

Public Property SequenceName() As String

###### Property Value

The name of the completed sequence.

###### Remarks

### SerialNumber Property

#### Reports the serial number of the DUT for the completed sequence.

##### Syntax

###### Declaration

Public Property SerialNumber() As String

###### Property Value

The serial number of the DUT for the completed sequence.

###### Remarks

# ObjectRepository Class

## Background

*TrueTestEngine Namespace*

The ObjectRepository is a thread-safe hashtable that can store any object for later retrieval. It is primarily used to pass information between individual Analyses within a Sequence at runtime. For instance, Analysis 1 and Analysis 2 may perform the same calculation at runtime. Analysis 2 may not know whether the user’s sequence ever contained an Analysis 1. But if it did, Analysis 1 can write the calculation to the ObjectRepository, and Analysis 2 can check the ObjectRepository to see if its calculation exists before spending time to do the calculation.

The ObjectRepository is cleared when the user chooses to run a full sequence (RunAll). Running a single Analysis step inside a sequence does not clear the Repository.

Items in the ObjectRepository are retrieved through a Key object, usually a simple object that does not take up much memory. If you have the same Key object that was used to Add an item, you can Get the item using the Key.

## Public Enums

### ReadWriteEnum

#### Used with GetItem. Sets whether or not code will only try to Read from the Repository, or will Write if the requested item wasn’t found.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| ReadOnlyNoWait | 0 | Tries to read an item without waiting to see if another thread is trying to write the item. |
| ReadOnlyWaitForWrite | 1 | Tries to read an item, but will wait if another thread has flagged that it will write the requested item |
| Write | 2 | Used to flag that an item will be written if it was not already found |

## Public Methods

### AddItem Method

#### Adds an item to the repository, using the specified key.

##### Syntax

###### Declaration

Public Shared Sub AddItem(*Key* As Object, *Item* As Object)

###### Parameters

*Key*

The object to use as the key for retrieving this item.

*Item*

The object to save in the repository for later retrieval.

###### Remarks

If an item exists in the repository with the same Key, it will get overwritten with the new Item.

### GetItem Method

#### Gets an item from the repository, using the specified key.

##### Syntax

###### Declaration

Public Shared Function GetItem(*Key* As Object, ByRef *ReadWrite* As TrueTestEngine.ObjectRepository.ReadWriteEnum) As Object

###### Parameters

*Key*

The object to use as the key for retrieving this item.

*ReadWrite*

Sets whether or not .

###### Returns

The object that had been stored in the Repository.

###### Remarks

If running TrueTest in Synchronous mode, using AddItem to add and setting ReadWrite = ReadOnlyNoWait is sufficient to retrieve. If running TrueTest in Asynchronous mode, there are times when you should set ReadWrite = Write and ReadWrite = ReadOnlyWaitForWrite. The usage pattern for asynchronous use of the ObjectRepository is as follows. Note that this is used according to the example given in the Background for this class: two Analyses want to save processing time and share a calculation, but do not know if the other analysis has completed the calculation or not.

1. Try to GetItem with ReadWrite = Write. If the item exists, you are done, as some other process has written (or is in the process of writing; see step 3) the object to the repository already.
2. If the item does not exist in the repository, GetItem will return an empty object of type ValueItem. Check for this by seeing if: Object.GetType.Name = “ValueItem”. If this expression returns true, then the Repository has signaled that it is waiting for you to write the object.
3. At this time, any other call to GetItem with ReadWrite = Write or ReadWrite = ReadOnlyWaitForWrite will cause the calling thread to wait until the requested object is written. The ReadWrite parameter is passed ByRef, and will also be set to WaitingToRead at this time.
4. Create the object you want to write.
5. Add the item with AddItem. Be sure to use the same Key used in step 1.
6. Any other calls to GetItem (see step 3) will immediately return the object added in step 5.

# PatternMeasurement Class

## Background

*TrueTestEngine Namespace*

The PatternMeasurement class contains the original camera measurement object.

Inside an Analysis, a programmer will have access to a list of PatternMeasurements that are passed as inputs. They can then pull out the original measurement, process it, and optionally save those measurements back into the PatternMeasurement object. These measurements, like LocalContrast or SyntheticMeasurement, are used in the TrueTest Report Generator.

## Public Properties

### CurrentMeasurement Property

#### The current index of the analysis within the Sequence class list Items. Can be used to know which step in the sequence is currently executing.

##### Syntax

###### Declaration

Public Property CurrentMeasurement() As RadiantCommon.MeasurementF

###### Property Value

The current working measurement. This measurement should have no processing applied.

###### Remarks

### CameraMeasurementID Property (Read Only)

#### Contains the unique ID number of the measurement in the database (\*.ttxm). If this is a new camera measurement that has not yet been written to the database, 0 will be returned.

##### Syntax

###### Declaration

Public ReadOnly Property CameraMeasurementID() As Integer

###### Property Value

The database ID of the measurement.

###### Remarks

This property is 0 when TrueTest is set to Use Camera, and nonzero when in Database mode (either Auto or User Select).

### BlobDrawShape Property

#### Selects a shape to draw for any blobs inside the DefectList.

##### Syntax

###### Declaration

Public Property BlobDrawShape() As RadiantCommon.RiBitmapCtl.BlobDrawShapeEnum

###### Property Value

The selected option for how to draw blobs from the DefectList.

###### Remarks

Only needed if the Analysis that has operated or will operate on this PatternMeasurement is set up to find blobs/defects. This information is used when drawing blobs in the user interface.

### DefectList Property

#### Contains a list of defect objects to be shown to the user.

##### Syntax

###### Declaration

Public Property DefectList() As List(Of RadiantCommon.DefectBase)

###### Property Value

A list of defect information generated by the analysis.

###### Remarks

Only needed if the Analysis that has operated, or will operate, on this PatternMeasurement is set up to find blobs/defects.

### LocalContrast Property

#### Contains a processed Local Contrast measurement based on the original camera measurement.

##### Syntax

###### Declaration

Public Property LocalContrast() As RadiantCommon.MeasurementF

###### Property Value

A processed measurement with units of local contrast.

###### Remarks

This measurement is optional and may not be set, depending on the Analysis.

### PatternSetup Property

#### The PatternSetup object used to take the original camera measurement.

##### Syntax

###### Declaration

Public Property PatternSetup() As TrueTestEngine.PatternSetup

###### Property Value

A PatternSetup object. See the PatternSetup class in this document for more info.

###### Remarks

The PatternSetup object contains all relevant information to be able to take another camera measurement in the same manner as CurrentMeasurement was taken.

### RADA Property

#### RADA stands for Register Active Display Area. For a rectangular device under test, a RADA algorithm can be applied that Registers the location of the active display area within the camera’s field of view. If this registration process has been performed, a rectangular measurement object is created removing any inactive border around the device under test, and saved to this property.

##### Syntax

###### Declaration

Public Property RADA() As RadiantCommon.MeasurementF

###### Property Value

A Registered rectangular measurement. This measurement represents only the active area of a device under test.

###### Remarks

This measurement is optional and may not be set, depending on the Analysis.

### Reference Property

#### In some defect processing Analyses, the original camera measurement is compared to a Reference image created from the original measurement to highlight defects. This property stores that Reference image for later review.

##### Syntax

###### Declaration

Public Property Reference() As RadiantCommon.MeasurementF

###### Property Value

A measurement object used in defect processing. This measurement typically appears very smooth, in contrast to the original measurement which may have defects.

###### Remarks

This measurement is optional and may not be set, depending on the Analysis.

### ROIArray Property

#### An array of Regions Of Interest (ROIs) to be displayed to the user. ROIs can be circles, rectangles, ellipses, or polygons.

##### Syntax

###### Declaration

Public Property ROIArray() As RadiantCommon.RegionOfInterest()

###### Property Value

The array of RegionOfInterest objects to be passed along to the user interface.

###### Remarks

An Analysis may want to draw rectangles, circles, etc to help the user understand what processing the Analysis is performing. This information can be retrieved through the ROIArray property after processing.

### SerialNumber Property

#### The serial number of the unit whose images and data are contained in this PatternMeasurement.

##### Syntax

###### Declaration

Public Property SerialNumber() As String

###### Property Value

A string serial number for the device that was measured.

###### Remarks

### SyntheticMeasurement Property

#### A synthetic measurement generated from data inside the original measurement. This measurement is most often a downsampling of data from the original measurement in a grid pattern, such as averaging data from a grid of Regions Of Interest.

##### Syntax

###### Declaration

Public Property SyntheticMeasurement() As RadiantCommon.MeasurementF

###### Property Value

A synthetic measurement generated by analysis.

###### Remarks

This measurement is optional and may not be set, depending on the Analysis.

# PatternSetup Class

## Background

*TrueTestEngine Namespace*

A PatternSetup object contains all information needed in order to take a measurement. The user defines a PatternSetup in the main user interface of TrueTest, and selects it together with an Analysis object to define one step in the Sequence.

## Constructors

### New

#### Initializes a new instance of the PatternSetup class.

##### Syntax

###### Declaration

Public Sub New()

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*NewName* as String)

**Parameters**

*NewName*

The name of the new pattern setup

###### Remarks

## Public Methods

### CloneMe Method

#### Clones the instance of the PatternSetup.

##### Syntax

###### Declaration

Public Function CloneMe(Optional *regenerateGUID* As Boolean) As PatternSetup

###### Return Value

Clone of the current PatternSetup

**Parameters**

*regenerateGUID*

If set to True this will create a new GUID for the measurement setup

###### Remarks

### GetMeasurementSetup Method

#### Gets the measurement setup object. Used if you want to take measurements through the PMEngine API instead of TrueTestEngine API. Note that TrueTest saves some additional metadata that PMEngine is unaware of. For the TrueTest Take Measurement function, see the TrueTest class.

##### Syntax

###### Declaration

Public Function GetMeasurementSetup() as PMEngine.MeasurementSetup

###### Return Value

The measurement setup object used for PMEngine.

###### Remarks

### SetFromMeasurementSetup Method

#### Sets the properties of a PatternSetup object based on a PMEngine.MeasurementSetup object.

##### Syntax

###### Declaration

Public Sub SetFromMeasurementSetup(*ms* as PMEngine.MeasurementSetup)

###### Parameters

*ms*

A measurement setup object. Its parameters will be filled into the current PatternSetup.

###### Remarks

## Public Properties

### ApplyMedianSmoothing Property

#### True or False whether to apply a median smoothing filter after taking the measurement.

##### Syntax

###### Declaration

Public Property ApplyMedianSmoothing() As Boolean

###### Property Value

True or False whether to apply Median filter

###### Remarks

Use together with MedianBlockSize property. Default and recommended value is False.

### AutoAdjustExposure Property

#### True or False whether to automatically adjust the exposure of the camera. If false, the software will use fixed exposure times.

##### Syntax

###### Declaration

Public Property AutoAdjustExposure() As Boolean

###### Property Value

True or False whether to automatically adjust exposure

###### Remarks

### BinningX Property

#### Gets or sets the binning in the x direction (horizontal binning).

##### Syntax

###### Declaration

Public Property BinningX() As Integer

###### Property Value

The horizontal binning to be used when a measurement is taken.

###### Remarks

### BinningY Property

#### Gets or sets the binning in the y direction (vertical binning).

##### Syntax

###### Declaration

Public Property BinningY() As Integer

###### Property Value

The vertical binning to be used when a measurement is taken.

###### Remarks

### CameraFlip Property

#### Gets or sets whether the measurement will be flipped.

##### Syntax

###### Declaration

Public Property CameraFlip() As Boolean

###### Property Value

Property indicates whether the measurement will be flipped.

###### Remarks

### CameraMirror Property

#### Gets or sets whether the measurement will be mirrored.

##### Syntax

###### Declaration

Public Property CameraMirror() As Boolean

###### Property Value

Property indicates whether the measurement will be mirrored.

###### Remarks

### CameraRotation Property

#### Gets or sets whether the measurement will be rotated.

##### Syntax

###### Declaration

Public Property CameraRotation() As TrueTest.CameraRotationEnum

###### Property Value

Property indicates whether the measurement will be rotated.

*The degrees to rotate the measurement*

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | No rotation applied |
| Clockwise90 | 90 | Measurement rotated 90 degrees |
| Rotate180 | 180 | Measurement rotated 180 degrees |
| Counterclockwise90 | 270 | Measurement rotated 270 degrees |

###### Remarks

This property will properly set the Flip, Mirror, Transpose properties to achieve the desired rotation. It is recommended that this property be used instead of Flip, Mirror, and Transpose.

### CameraTranspose Property

#### Gets or sets whether the measurement will be transposed.

##### Syntax

###### Declaration

Public Property Transpose() As Boolean

###### Property Value

Property indicates whether the measurement will be transposed.

###### Remarks

### CaptureFilter Property

#### Gets or sets whether or not to capture an image for each camera filter

##### Syntax

###### Declaration

Public Property CaptureFilter(*index* as integer) As Boolean

###### Parameters

*index*

The filter index. Typically, Y filter = 1, X filter = 2, Z filter = 3. If optional XB filter is installed, XB = 4. Positions 0, 5 and 6 are unused.

###### Property Value

Whether or not to capture an image at the requested filter index

###### Remarks

This property is indexed, which could require unusual syntax in languages other than VB.NET.

### ConoscopeCalibrationID Property

#### Gets or sets the conoscope calibration ID.

##### Syntax

###### Declaration

Public Property ConoscopeCalibrationID() As Integer

###### Property Value

The database ID number of the conoscope calibration to be used.

###### Remarks

### DistanceUnit Property

#### Gets or sets the distance unit to be applied to measurements.

##### Syntax

###### Declaration

Public Property DistanceUnit() As RadiantCommon.DistanceUnitType

###### Property Value

Property indicates the distance unit to be used.

*DistanceUnitType Enum*

See Common Enumerations – RadiantCommon Namespace at the end of this document for enumeration listing.

###### Remarks

### DUTDistance Property

#### Gets or sets the distance of the DUT (device under test) to the camera in meters.

##### Syntax

###### Declaration

Public Property DUTDistance() As Single

###### Property Value

Property indicates the DUT distance in meters.

###### Remarks

Setting this value is only necessary if the measurement was an illuminance measurement that will be converted into a luminous intensity measurement. (MakeIntensity property must be set to True.)

### ExposureLimitMethod Property

#### Gets or sets the method for finding the exposure time

##### Syntax

###### Declaration

Public Property ExposureLimitMethod() As RadiantVisionSystems.Camera.CameraEnumerators.ExposureLimitType

###### Property Value

The exposure limit method

*ExposureLimitType Enum*

The spectral response of a measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | Uses Standard method (histogram). Doesn’t show a message box if overexposed |
| Standard | 1 | Uses Standard method (histogram). Shows a message box if overexposed. |
| BrightSpot | 2 | Uses the BrightSpot method when adjusting exposure or taking a measurement (message box shows if overexposed) |
| BrightSpot2x2 | 3 | Uses the BrightSpot2x2 method when adjusting exposure or taking a measurement (message box shows if overexposed) |
| BrightSpot3x3 | 4 | Uses the BrightSpot3x3 method when adjusting exposure or taking a measurement (message box shows if overexposed) |
| BrightSpot4x4 | 5 | Uses the BrightSpot4x4 method when adjusting exposure or taking a measurement (message box shows if overexposed) |

###### Remarks

### ExposureShowOverLimitMessage Property

#### Gets or sets whether or not to show a message box if a measurement taken with this setup has overexposed

##### Syntax

###### Declaration

Public Property ExposureShowOverLimitMessage() As Boolean

###### Property Value

True or False, whether or not to show a message box if a measurement has overexposed

###### Remarks

By default this should be false, as a message box will stop the rest of the application from progressing.

### ExposureTime Property

#### Gets or sets the exposure time in milliseconds for each camera filter

##### Syntax

###### Declaration

Public Property ExposureTime(*index* as integer) As Single

###### Parameters

*index*

The filter index. Typically, Y filter = 1, X filter = 2, Z filter = 3. If optional XB filter is installed, XB = 4. Positions 0, 5 and 6 are unused.

###### Property Value

The exposure time in milliseconds at the requested filter index

###### Remarks

This property is indexed, which could require unusual syntax in languages other than VB.NET.

### GoniometerType Property

#### Gets or sets the goniometer type used for creating a luminous intensity measurement from an illuminance measurement

##### Syntax

###### Declaration

Public Property GoniometerType() As RadiantCommon.GoniometerType

###### Property Value

The goniometer type to use for transformation into intensity

*GoniometerType Enum*

The spectral response of a measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| SIG | 0 | Source Imaging Goniometer – a goniometer device produced by Radiant Vision Systems. A left-handed version of TypeC. |
| TypeA | 1 | Type A goniometer |
| TypeB | 2 | Type B goniometer (H/V) |
| TypeC | 3 | Type C goniometer (theta/phi) |
| ProjectedAngles | 4 | Project angles from DUT onto image – an approximation method |

###### Remarks

Only used if MakeIntensity is set to True.

### GUID Property

#### Gets or sets the unique ID associated with this setup

##### Syntax

###### Declaration

Public Property GUID() As String

###### Property Value

The globally unique identifier for this setup

###### Remarks

This property is automatically set by TrueTest when the setup is first created.

### HDRImagingEnabled Property

#### Gets or sets whether to use HDR (high dynamic range) imaging.

##### Syntax

###### Declaration

Public Property HRDImagingEnabled() As Boolean

###### Property Value

True or False, whether to use HDR imaging.

###### Remarks

### LEDColorCalibration Property

#### Gets or sets whether to use the LED color calibration.

##### Syntax

###### Declaration

Public Property LEDColorCalibration() As Boolean

###### Property Value

True or False, whether to use LED color calibration.

###### Remarks

### LensDistance Property

#### Gets or sets the focus distance for the lens

##### Syntax

###### Declaration

Public Property LensDistance() As Single

###### Property Value

The focus distance for the lens, as measured from the camera’s faceplate to the DUT

###### Remarks

Also called working distance. Only used for cameras that support software-controllable lenses.

### LensfNumber Property

#### Gets or sets the f-number to use on the lens.

##### Syntax

###### Declaration

Public Property LensfNumber() As String

###### Property Value

The lens f-number to use for this setup

###### Remarks

If an illegal value is used to set the f-number, the aperture will be positioned wide open.

### MakeIntensity Property

#### Gets or sets whether or not to transform an illuminance measurement into a luminous intensity measurement.

##### Syntax

###### Declaration

Public Property MakeIntensity() As Boolean

###### Property Value

True or False, whether or not to make a luminous intensity measurement

###### Remarks

Only used if the Flat Field for this setup was made for illuminance (instead of luminance). Other properties will need to be set correctly for the intensity measurement to be made properly (GoniometerType, DUTDistance).

### MeasurementSetupID Property

#### Gets or sets the measurement setup ID

##### Syntax

###### Declaration

Public Property MeasurementSetupID() As Integer

###### Property Value

The measurement setup ID

###### Remarks

Only used when writing to the database.

### MedianBlockSize Property

#### Gets or sets the block size (in pixels) used if performing a median filter operation

##### Syntax

###### Declaration

Public Property MedianBlockSize() As Integer

###### Property Value

The size in image pixels of the neighborhood block size to use for a median filter

###### Remarks

Only used if ApplyMedianSmoothing is True.

### Name Property

#### Gets or sets the pattern setup name. This name is required to be unique among all Pattern Setup names in the current Sequence.

##### Syntax

###### Declaration

Public Property Name() As String

###### Property Value

The name of the pattern setup

###### Remarks

### NbrCaptureFilters Property (Read Only)

#### Gets the number of filters used in the measurement

##### Syntax

###### Declaration

Public ReadOnly Property NbrCaptureFilters() As Integer

###### Property Value

The number of filters used in the measurement

###### Remarks

### NbrDarkFramesToAverage Property

#### Gets or sets the number of dark frames to average

##### Syntax

###### Declaration

Public Property NbrDarkFramesToAverage() As Integer

###### Property Value

The number of dark frames to average

###### Remarks

### NbrFramesToAverage Property

#### Gets or sets the number of measurements to average

##### Syntax

###### Declaration

Public Property NbrFramesToAverage() As Integer

###### Property Value

The number of measurements to average

###### Remarks

### NDFilterPosition Property

#### Gets or sets the ND filter position to use for each camera filter

##### Syntax

###### Declaration

Public Property NDFilterPosition(*index* as integer) As Integer

###### Parameters

*index*

The filter index. Typically, Y filter = 1, X filter = 2, Z filter = 3. If optional XB filter is installed, XB = 4. Positions 0, 5 and 6 are unused.

###### Property Value

The ND filter index to use at the requested color filter index. Typically ND0 (clear) = 1, ND1 = 2, ND2 = 3, etc.

###### Remarks

This property is indexed, which could require unusual syntax in languages other than VB.NET.

### Notes Property

#### Gets or sets the notes on the pattern setup

##### Syntax

###### Declaration

Public Property Notes() As String

###### Property Value

The notes on the pattern setup

###### Remarks

### Pattern Property

#### Gets or sets the display pattern used for the measurement

##### Syntax

###### Declaration

Public Property Pattern() As TrueTestPatternGenerator.TrueTestPattern

###### Property Value

The pattern used for the measurement

###### Remarks

### PhotometricUnit Property

#### Gets or sets the photometric quantity associated with this setup

##### Syntax

###### Declaration

Public Property PhotometricUnit() As RadiantCommon.PhotometricUnitType

###### Property Value

The unit associated with this measurement (e.g. cd/m2 for luminance, lux for illuminance)

###### Remarks

See the Common Enumerations list at the end of this document for more information about the PhotometricUnitType enum.

### SpectralResponse Property

#### Gets or sets the spectral response type for this measurement

##### Syntax

###### Declaration

Public Property SpectralRespone() As RadiantCommon.SpectralResponseType

###### Property Value

The spectral response for this measurement (0 = Radiometric, 1 = Photometric)

###### Remarks

Almost all measurements are Photometric. Radiometric measurements require a radiometric filter to be installed in the camera.

### ThresholdFilter Property

#### Gets or sets the threshold to apply (as a %) for each camera filter

##### Syntax

###### Declaration

Public Property ThresholdFilter(*index* as integer) As Single

###### Parameters

*index*

The filter index. Typically, Y filter = 1, X filter = 2, Z filter = 3. If optional XB filter is installed, XB = 4. Positions 0, 5 and 6 are unused.

###### Property Value

The % of max threshold to apply at the requested filter index. Each filter image is processed independently.

###### Remarks

This property is indexed, which could require unusual syntax in languages other than VB.NET.

# Result Class

## Background

*TrueTestEngine Namespace*

The various analyses call this class to record measurement and analysis results. Results can be optionally given pass/fail criteria, which are then used to determine if the Analysis as a whole has passed or failed.

## Constructors

### New

#### Initializes a new instance of the Result class.

##### Syntax

###### Declaration

Public Sub New()

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*Analysis* as TrueTestEngine.AnalysisBase)

**Parameters**

*Analysis*

The analysis related to the result

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*Analysis* as TrueTest.AnalysisBase, *Name* as String, *Value* as Single, *Unit* as String, Optional *PassFail* as TrueTestEngine.TrueTest.AnalysisResultEnum = TrueTestEngine.TrueTest.AnalysisResultEnum.None )

**Parameters**

*Analysis*

The analysis related to the result

*Name*

Name of the result

*Value*

Value of the result as Single

*Unit*

Unit used to measure the result

*PassFail*

Optional parameter indicating if the result passed or failed test parameters

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*Analysis* as TrueTest.AnalysisBase, *Name* as String, *Value* as String, *Unit* as String, Optional *PassFail* as TrueTestEngine.TrueTest.AnalysisResultEnum = TrueTestEngine.TrueTest.AnalysisResultEnum.None )

**Parameters**

*Analysis*

The analysis related to the result

*Name*

Name of the result

*Value*

Value of the result as String

*Unit*

Unit used to measure the result

*PassFail*

Optional parameter indicating if the result passed or failed test parameters

###### Remarks

## Public Properties

### AnalysisName Property

#### The name of the analysis

##### Syntax

###### Declaration

Public Property AnalysisName() As String

###### Property Value

The name of the analysis

###### Remarks

### Name Property

#### The name of the result.

##### Syntax

###### Declaration

Public Property Name() As String

###### Property Value

The name of the result

###### Remarks

### PassFail Property

#### Enumeration indicating whether the result passed or failed

##### Syntax

###### Declaration

Public Property PassFail() As TrueTest.AnalysisResultEnum

###### Property Value

Enumeration indicating whether the result passed or failed

Analysis Result Enum

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | No pass or fail applies |
| Pass | 1 | The result passed |
| Fail | 2 | The result failed |

###### Remarks

### PatternName Property

#### The name of the pattern used during measurement.

##### Syntax

###### Declaration

Public Property PatternName() As String

###### Property Value

The name of the pattern used during analysis

###### Remarks

### SequenceName Property

#### The name of the sequence where this result came from.

##### Syntax

###### Declaration

Public Property SequenceName() As String

###### Property Value

The name of the sequence used

###### Remarks

### SerialNumber Property

#### The DUT (device under test) serial number associated with the result.

##### Syntax

###### Declaration

Public Property SerialNumber() As String

###### Property Value

The serial number associated with the result

###### Remarks

### Unit Property

#### The unit of the result value (mm, cd/m2, etc)

##### Syntax

###### Declaration

Public Property Unit() As String

###### Property Value

The unit of the result value

###### Remarks

### Value Property

#### The value of the result reported numerically

##### Syntax

###### Declaration

Public Property Value() As Double

###### Property Value

The value of the result

###### Remarks

### ValueString Property

#### The value of the result reported as a string

##### Syntax

###### Declaration

Public Property ValueString() As string

###### Property Value

The value of the result as a string

###### Remarks

# Sequence Class

## Background

*TrueTestEngine Namespace*

The Sequence class contains all information that will be saved to the Sequence definition file (\*.seqx). Primarily, it is a list of SequenceItems that corresponds to the same list of tests shown to the user. It also is where commands to start testing are received.

Whenever possible, you should start the Sequence using commands from the TrueTest class rather than calling a Sequence directly.

## Constructors

### New

#### Initializes a new instance of the Sequence class.

##### Syntax

###### Declaration

Public Sub New()

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*SetupXMLFilePathName* as String)

**Parameters**

*SetupXMLFilePathName*

The file path used to save the XML data

###### Remarks

## Public Methods

### Clone Method

#### Makes a clone copy of the current sequence

##### Syntax

###### Declaration

Public Function Clone() as TrueTestEngine.Sequence

###### Return Value

A clone copy of the sequence

###### Remarks

### GetPatternSetupByName Method

#### Gets the pattern setup object using the name

##### Syntax

###### Declaration

Public Function GetPatternSetupByName(*PatternSetupSearchName* as String) as TrueTestEngine.PatternSetup

###### Parameters

*PatternSetupSearchName*

Name of the pattern setup

###### Return Value

The pattern setup object

###### Remarks

### GetSeqItemByGUID Method

#### Gets the full SequenceItem (including Analysis object) in the sequence associated with a string GUID.

##### Syntax

###### Declaration

Public Function GetSeqItemByGUID(*GUID* as String) as SequenceItem

###### Parameters

*GUID*

String representation of the GUID for the sequence item to be retrieved.

###### Return Value

Sequence item associated with the GUID

###### Remarks

### GetSeqItemByGUID Method

#### Gets the full SequenceItem (including Analysis object) in the sequence associated with a string GUID. This overload requires an empty SequenceItem to be passed in by reference.

##### Syntax

###### Declaration

Public Function GetSeqItemByGUID(*GUID* as String, ByRef *SequenceItem* as SequenceItem)

###### Parameters

*GUID*

String representation of the GUID for the sequence item to be retrieved.

*SequenceItem*

The empty SequenceItem class to be filled and returned.

###### Remarks

### GetSequenceItemByPatternSetupName Method

#### Gets the full SequenceItem (including Analysis object) in the sequence associated with the pattern setup name.

##### Syntax

###### Declaration

Public Function GetSequenceItemByPatternSetupName(*PatternSetupSearchName* as String) as List(of SequenceItem)

###### Parameters

*PatternSetupSearchName*

Name of the pattern setup

###### Return Value

List of sequence items associated with the pattern setup name

###### Remarks

### LoadComboBoxWithPatterns Method

#### Gets a list of pattern setups associated with the analyses in the sequence

##### Syntax

###### Declaration

Public Sub LoadComboBoxWithPatterns(ByRef *ComboBox* as ComboBox, Optional *SelectedPattern* As String = “”)

###### Parameters

*ComboBox*

Combo box to load the patterns

*SelectedPattern*

Pattern selected after the combobox is loaded

###### Remarks

### SetPatternSetup Method

#### Updates a pattern setup with values from a new pattern setup. The first pattern setup with the same name as the new pattern setup will get replaced.

##### Syntax

###### Declaration

Public Sub SetPatternSetup(*NewPatternSetup* As TrueTestEngine.PatternSetup)

###### Parameters

*NewPatternSetup*

The pattern setup to use

###### Remarks

### TestTimeSec Method

#### Retrieves the total test time (in seconds) of the most recently run sequence, or 0 if no test time is available.

##### Syntax

###### Declaration

Public Function TestTimeSec() As Single

###### Return Value

The number of seconds elapsed during the most recent sequence run.

###### Remarks

## Public Properties

### CurrentSequenceIndex Property

#### The index of the current analysis being run.

##### Syntax

###### Declaration

Public Property CurrentSequenceIndex() As Integer

###### Property Value

Integer

###### Remarks

### Description Property

#### The description for the sequence, which is the sequence filename.

##### Syntax

###### Declaration

Public Property Description() As String

###### Property Value

String

###### Remarks

If the filename is ‘MySequence.seqx’, the Description is ‘MySequence’.

### IsComplete Property

#### True or False indicating if the sequence has been completed

##### Syntax

###### Declaration

Public Property IsComplete() As Boolean

###### Property Value

True or False

###### Remarks

### Items Property

#### The heart of a sequence is its Items list, containing the list of steps to be executed.

##### Syntax

###### Declaration

Public Property Items() As List(Of TrueTestEngine.SequenceItem)

###### Property Value

A list of items in the sequence

###### Remarks

### MasterDefectList Property

#### A list of all the defects found from all analyses performed in the sequence.

##### Syntax

###### Declaration

Public Property MasterDefectList() As TrueTestEngine.MasterDefectList

###### Property Value

List of defects

###### Remarks

### Name Property (Read Only)

#### The name of the sequence.

##### Syntax

###### Declaration

Public ReadOnly Property Name() As String

###### Property Value

The name of the sequence. This is the filename, without the \*.seqx extension.

###### Remarks

### NumFailedTests Property (Read Only)

#### Number of failed tests in the sequence.

##### Syntax

###### Declaration

Public ReadOnly Property NumFailedTests() As Integer

###### Property Value

Number of failed tests

###### Remarks

### NumPassedTests Property (Read Only)

#### Number of passed tests in the sequence.

##### Syntax

###### Declaration

Public ReadOnly Property NumPassedTests() As Integer

###### Property Value

Number of passed tests

###### Remarks

### PassedSecurity Property

#### True or False indicating if the software has passed security. Security checks the license code against the key.

##### Syntax

###### Declaration

Public Shared Property PassedSecurity() As Boolean

###### Property Value

True or False

###### Remarks

Security is checked once per sequence run.

### PassFail Property

#### Indicates whether the sequence has passed or failed.

##### Syntax

###### Declaration

Public Property PassFail() As TrueTest.AnalysisResultEnum

###### Property Value

Pass or Fail

Analysis Result Enum

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | No pass or fail applies |
| Pass | 1 | The result passed |
| Fail | 2 | The result failed |

###### Remarks

### PatternGen Property

#### The pattern generator used. This property is only used if the global Settings property ReadPGPropertiesFromSeqxFile is set to True. Otherwise it is null (nothing).

##### Syntax

###### Declaration

Public Property PatternGen() As TrueTestPatternGenerator.PatternGeneratorBaseWrapper

###### Property Value

TrueTestPatternGenerator.PatternGeneratorBaseWrapper

###### Remarks

### PatternSetupList Property

#### List of available pattern setups in the sequence definition.

##### Syntax

###### Declaration

Public Property PatternSetupList() As List(Of PatternSetup)

###### Property Value

List of pattern setups

###### Remarks

### SequenceStopped Property (Read Only)

#### True or False indicating if the sequence has been stopped/cancelled

##### Syntax

###### Declaration

Public ReadOnly Property SequenceStopped() As Boolean

###### Property Value

True or False

###### Remarks

### SerialNumber Property

#### Serial number of the device under test.

##### Syntax

###### Declaration

Public Property SerialNumber() As String

###### Property Value

String

###### Remarks

### XMLFilePathName Property

#### Current File path definition for the sequence. Also used to save the sequence.

##### Syntax

###### Declaration

Public Property XMLFilePathName() As String

###### Property Value

File path as String

###### Remarks

# SequenceItem Class

## Background

*TrueTestEngine Namespace*

A SequenceItem is one step in the sequence, containing both a desired Analysis and the PatternSetup used to create the Measurement to be analyzed.

## Public Properties

### Analysis Property

#### Analysis associated with this item in the sequence.

##### Syntax

###### Declaration

Public Property Analysis() As TrueTestEngine.AnalysisBase

###### Property Value

Analysis associated with this item in the sequence

###### Remarks

### GUID Property

#### Globally unique identifier associated with this item in the sequence.

##### Syntax

###### Declaration

Public Property GUID() As String

###### Property Value

GUID string associated with this item in the sequence

###### Remarks

The GUID is automatically generated when the sequence item is first created.

### Measurement Property (Read Only)

#### Measurement associated with this item in the sequence

##### Syntax

###### Declaration

Public ReadOnly Property Measurement() As PMEngine.PMMeasurementF

###### Property Value

Measurement object

###### Remarks

### PatternSetupName Property

#### Name of the pattern setup associated with this item in the sequence

##### Syntax

###### Declaration

Public Property PatternSetupName() As String

###### Property Value

String name of the PatternSetup object selected for this step in the sequence.

###### Remarks

### Selected Property

#### True or False indicating if the item in the sequence has been selected for analysis.

##### Syntax

###### Declaration

Public Property Selected() As Boolean

###### Property Value

True or False indicating if the item has been selected for analysis.

###### Remarks

# Settings Class

## Background

*TrueTestEngine Namespace*

The Settings class contains globally relevant settings to apply when running a sequence, such as whether or not to run asynchronously, which types of measurements to save to the database, etc.

## Constructors

### New

#### Initializes a new instance of the Settings class.

##### Syntax

###### Declaration

Public Sub New()

###### Remarks

## Public Methods

### Save

#### Saves the current settings.

##### Syntax

###### Declaration

Public Sub Save()

###### Remarks

## Public Properties

### AnalysisSequenceMode Property

#### Mode used to run the analyses. Asynchronous mode makes use of the multithreading features of TrueTest.

##### Syntax

###### Declaration

Public Property AnalysisSequenceMode() As TrueTest.SequenceModeEnum

###### Property Value

Mode used to run analyses. Options are Synchronous or Asynchronous.

###### Remarks

### AnalysisTestSelectionInOperatorMode Property

#### Allows the ‘Select’ checkbox of the Sequence Control user interface to appear even in Operator Mode. This means operators can choose to ignore certain tests by deselecting them before running the sequence.

##### Syntax

###### Declaration

Public Property AnalysisTestSelectionInOperatorMode() As Boolean

###### Property Value

Whether or not to show ‘Select’ checkboxes next to each step in a sequence when in Operator mode.

###### Remarks

### AutoGenerateDatabase Property

#### Selects whether or not to automatically generate a new measurement database (\*.ttxm) under the following conditions: once daily, and if the current database size is greater than 2.5 GB.

##### Syntax

###### Declaration

Public Property AutoGenerateDatabase() As Boolean

###### Property Value

Whether or not to automatically generate a measurement database daily or if the current file size is greater than 2.5GB.

###### Remarks

### CheckDuplicateSerialNumber Property

#### Checks if the serial number has already been used.

##### Syntax

###### Declaration

Public Property CheckDuplicateSerialNumber() As Boolean

###### Property Value

True or False

###### Remarks

### DatabaseModeDelay Property

#### Delay (in milliseconds) after each measurement is read from the database.

##### Syntax

###### Declaration

Public Property DatabaseModeDelay() As Integer

###### Property Value

Milliseconds to delay after each measurement is read from the database

###### Remarks

Used primarily for demo purposes in Database mode to simulate the time it takes to measure with a camera even if no camera is present.

### DisplayCameraMeasurements Property

#### Whether or not to display original camera measurements to the user.

##### Syntax

###### Declaration

Public Property DisplayCameraMeasurements() As Boolean

###### Property Value

True or False.

###### Remarks

### DisplayDefectList Property

#### Whether or not to display defect lists to the user.

##### Syntax

###### Declaration

Public Property DisplayDefectList() As Boolean

###### Property Value

True or False

###### Remarks

### DisplayMasterDefectList Property

#### Whether or not to display a master defect list to the user after a sequence completes.

##### Syntax

###### Declaration

Public Property DisplayMasterDefectList() As Boolean

###### Property Value

True or False

###### Remarks

Not all TrueTest Analyses add defects to the master list.

### DisplayUpdatedMeasurements Property

#### Whether or not to display the updated (processed) measurements during the test process to the user.

##### Syntax

###### Declaration

Public Property DisplayUpdatedMeasurements() As Boolean

###### Property Value

True or False

###### Remarks

### EnterUIinOperatorMode Property

#### If set to True, TrueTest will automatically start up in Operator Mode. By default (or if this value is set to False), TrueTest will start in Engineering Mode, allowing anyone to make changes.

##### Syntax

###### Declaration

Public Property EnterUIinOperatorMode() As Boolean

###### Property Value

Whether or not to automatically start up TrueTest in Operator Mode

###### Remarks

### FailColor Property

#### Color used to indicate a failed test

##### Syntax

###### Declaration

Public Property FailColor() As System.Drawing.Color

###### Property Value

System.Drawing.Color

###### Remarks

### MeasurementSequenceMode Property

#### Mode used to take the measurements. Asynchronous mode makes use of the multithreading features of TrueTest

##### Syntax

###### Declaration

Public Property MeasurementSequenceMode() As TrueTest.SequenceModeEnum

###### Property Value

Mode used to take the measurements. Options are Synchronous or Asynchronous.

###### Remarks

### PassColor Property

#### Color used to indicate a passed test

##### Syntax

###### Declaration

Public Property PassColor() As System.Drawing.Color

###### Property Value

Color

###### Remarks

### Password Property

#### Password used to enter the engineering view from the operator view

##### Syntax

###### Declaration

Public Property Password() As String

###### Property Value

String

###### Remarks

If you change this property in code, it will also affect the standard User Interface of TrueTest that runs on the same PC.

### PatternGenerator Property

#### Pattern generator selected for the sequence.

##### Syntax

###### Declaration

Public Property PatternGenerator() As TrueTestPatternGenerator.PatternGeneratorBaseWrapper

###### Property Value

The pattern generator used for testing.

###### Remarks

### ReadMSfromCalFile Property

#### Whether or not to read Measurement Setup information from the camera calibration file, instead of the sequence (\*.seqx) file. Measurement Setup information is always saved to both \*.seqx and \*.calx camera calibration file.

##### Syntax

###### Declaration

Public Property ReadMSfromCalFile() As Boolean

###### Property Value

True or False

###### Remarks

### ReadPGpropertiesFromSeqx Property

#### Whether or not to save/read Pattern Generator properties to the sequence (\*.seqx) file. If False, the pattern generator properties will be saved globally for all sequence files. If True, each sequence file can save its own pattern generator and pattern generator properties.

##### Syntax

###### Declaration

Public Property ReadPGpropertiesFromSeqx() As Boolean

###### Property Value

True or False

###### Remarks

### RegressionTest Property

#### If set to True, once the sequence is started, it will repeat continously until cancelled

##### Syntax

###### Declaration

Public Property RegressionTest() As Boolean

###### Property Value

True or False

###### Remarks

### SaveAnalysisResultsToDatabase Property

#### Whether or not to save the analysis results to the database.

##### Syntax

###### Declaration

Public Property SaveAnalysisResultsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveFFTMeasurementsToDatabase Property

#### True or False indicating whether to save the Fast Fourier Transform measurements to the database.

##### Syntax

###### Declaration

Public Property SaveFFTMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveFinalAnalysisImages Property

#### True or False indicating whether to save the final analysis images. These measurements contain overlays with potentially useful visualizations for the user.

##### Syntax

###### Declaration

Public Property SaveFinalAnalysisImages() As Boolean

###### Property Value

True or False

###### Remarks

### SaveFinalAnalysisImagesDelay Property

#### Delay for saving the final analysis images.

##### Syntax

###### Declaration

Public Property SaveFinalAnalysisImagesDelay() As Integer

###### Property Value

Delay for saving final analysis images, in ms.

###### Remarks

### SaveFinalAnalysisImagesPath Property

#### The location for saving the final analysis images.

##### Syntax

###### Declaration

Public Property SaveFinalAnalysisImagesPath() As String

###### Property Value

The location to save the final analysis images.

###### Remarks

### SaveFinalAnalysisMeasurementsToDatabase Property

#### True or False indicating whether to save the final analysis measurements to the database. These measurements contain overlays with potentially useful visualizations for the user.

##### Syntax

###### Declaration

Public Property SaveFinalAnalysisMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveLocalContrastMeasurementsToDatabase Property

#### True or False indicating whether to save the local contrast measurements to the database.

##### Syntax

###### Declaration

Public Property SaveLocalContrastMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveOnlyFailedMeasurements Property

#### True or False indicating whether to save only the failed measurements to the database.

##### Syntax

###### Declaration

Public Property SaveOnlyFailedMeasurements() As Boolean

###### Property Value

True or False

###### Remarks

### SaveReferenceMeasurementsToDatabase Property

#### True or False indicating whether to save the Reference measurements to the database.

##### Syntax

###### Declaration

Public Property SaveReferenceMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveRegisteredMeasurementsToDatabase Property

#### True or False indicating whether to save the registered measurements to the database.

##### Syntax

###### Declaration

Public Property SaveRegisteredMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SaveSyntheticMeasurementsToDatabase Property

#### True or False indicating whether to save the synthetic measurements to the database.

##### Syntax

###### Declaration

Public Property SaveSyntheticMeasurementsToDatabase() As Boolean

###### Property Value

True or False

###### Remarks

### SerialNumberMode Property

#### SerialNumberMode indicates when to enter the serial number of the device under test

##### Syntax

###### Declaration

Public Property SerialNumberMode() As TrueTestEngine.Settings.SerialNumberModeEnum

###### Property Value

Serial Number Mode Enum

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | No serial number |
| BeforeMeasurement | 1 | The software will request a serial number before the measurement |
| AfterMeasurement | 2 | The software will request a serial number after the measurement |

###### Remarks

### ShowAdvancedOptionsInPS Property

#### True or False indicating whether to show advanced setup options in the Pattern Setup user interface control.

##### Syntax

###### Declaration

Public Property ShowAdvancedOptionsInPS() As Boolean

###### Property Value

True or False

###### Remarks

### ShowStatusForm Property

#### True or False indicating whether to show the camera’s status form during the measurement taking process. This form shows more detailed status of the camera’s current actions.

##### Syntax

###### Declaration

Public Property ShowStatusForm() As Boolean

###### Property Value

True or False

###### Remarks

# PatternBase Class

## Background

*TrueTestPatternGenerator Namespace*

The PatternBase Class is a must inherit class for setting up patterns to send to the pattern generator.

## Public Methods

### CloneMe Method

#### Clones the instance of the PatternBase.

##### Syntax

###### Declaration

Public Function CloneMe() As TrueTestPatternGenerator.PatternBase

###### Return Value

Clone of the current PatternBase object.

###### Remarks

### ColorDescription Method

#### Returns a string name for a color based on the RGB values.

##### Syntax

###### Declaration

Protected Function ColorDescription(*Color* as System.Drawing.Color) As String

###### Parameters

###### *Color*

The color object to get the description as a string

###### Return Value

The string name of the color.

###### Remarks

### GenerateImage Method

#### Override this function in order to use this pattern with a VideoPatternGenerator

##### Syntax

###### Declaration

Protected Friend Overrideable Function GenerateImage(*Resolution* as System.Drawing.Size) As System.Drawing.Image

###### Parameters

###### *Resolution*

The resolution of the display

###### Return Value

The image object to be displayed

###### Remarks

If you will not use this pattern type with the VideoPatternGenerator, then you do not need to write this function. Simply have the function return Nothing.

### IsPatternEqualTo Method

#### Uses reflection to check the properties of the pattern and see if it is equal to another pattern. Override to ignore specific properties when checking equality. The primary reason to check for equality is to tell the user whether or not they need to save the pattern object if it has been changed.

##### Syntax

###### Declaration

Protected Overrideable Function IsPatternEqualTo(*pattern* as TrueTestPatternGenerator.PatternBase) As Boolean

###### Parameters

###### *Pattern*

The pattern used to check equality

###### Return Value

True or False indicating equality

###### Remarks

It is not necessary to override this function unless you want explicit control over how one pattern is considered equal to another.

## Public Properties

### Name Property

#### Gets or sets the pattern base name.

##### Syntax

###### Declaration

Public Property Name() As String

###### Property Value

The pattern base name

###### Remarks

# PatternGeneratorBase Class

## Background

*TrueTestPatternGenerator Namespace*

The PatternGeneratorBase class is a must-inherit class for pattern generator classes. The PatternGenerator will control all interaction between TrueTest and external hardware used to generate a desired pattern on the DUT (device under test). One simple example of a PatternGenerator is a PC video card, which can output a pattern to an external monitor, which would be the DUT.

NOTE: Override the Dispose and/or Finalize methods if your custom PatternGenerator calls external, unmanaged DLLs (e.g. C++ dlls) so that you can cleanly dispose of unmanaged memory.

## Public Events

### RefreshPropertyGrid Event

#### Occurs every time a request is made for the UI to update itself because PatternGeneratorBase properties were modified.

##### Syntax

###### Declaration

Public Shared Event RefreshPropertyGrid()

###### Remarks

It is generally not required to handle this event unless you are creating a custom control to view/edit Pattern Generator settings.

## Public and Protected Methods

### CloneMe Method

#### Clones the instance of the PatternGeneratorBase.

##### Syntax

###### Declaration

Public Function CloneMe() As TrueTestPatternGenerator.PatternGeneratorBase

###### Return Value

Clone of the current PatternGeneratorBase

###### Remarks

### GetCustomForm Method

#### Returns a custom form. Override this only if you also override HasCustomDeviceControlForm = True.

##### Syntax

###### Declaration

Protected Friend Overridable Function GetCustomForm() As System.Windows.Form

###### Return Value

The custom form

###### Remarks

### Initialize Method

#### Initializes the instance of PatternGeneratorBase

##### Syntax

###### Declaration

Protected Friend Overrideable Property Initialize() As Boolean

###### Return Value

True or False; whether or not the PatternGenerator was initialized.

###### Remarks

### IsInitializationRequired Method

#### True or False whether it is required to initialize the pattern generator

##### Syntax

###### Declaration

Protected Friend Overrideable Function IsInitializationRequired(*NewPGObject* as TrueTestPatternGenerator.PatternGeneratorBase) As Boolean

###### Parameters

###### *NewPGObject*

The new pattern generator object

###### Return Value

True or False indicating whether to initialize the new pattern generator object

###### Remarks

Only override if your PatternGenerator requires a special initialization routine to be performed.

### OnRefreshPropertyGrid Method

#### Used to raise an event to the user interface requesting that it update itself, as the underlying properties of the PatternGeneratorBase have changed. Use this method if creating properties that can modify other properties.

##### Syntax

###### Declaration

Protected Sub OnRefreshPropertyGrid()

###### Remarks

### SetBrowsableAttributes Method

#### Used with auto hide/show of properties in the PatternGenerator property grid. Sets initial conditions for properties to be browsable or not. Override if you want manual control over showing/hiding specific properties.

##### Syntax

###### Declaration

Public Overrideable Sub SetBrowsableAttributes()

###### Remarks

### SetPatternReady Method

#### If your PatternGenerator requires external code to control the device and signal that the pattern is ready, have that external code call this sub to set the PatternReady flag.

##### Syntax

###### Declaration

Protected Friend Sub SetPatternReady()

###### Remarks

### ShowPattern Method

#### Shows a pattern. To Hide a pattern, handle the case where p is nothing.

##### Syntax

###### Declaration

Protected Friend MustOverride Sub ShowPattern(*p* as TrueTestPatternGenerator.TrueTestPattern)

###### Parameters

*p*

The pattern to show.

###### Remarks

### ShowRGBWPattern Method

#### Shows a fullscreen red, green, blue, or white color pattern. This Sub is only called from the Four Color Calibration form. If your device does not have RGB primaries, you do not need to override this Sub.

##### Syntax

###### Declaration

Protected Friend Overridable Sub ShowRGBWPattern(*color* as Integer)

###### Parameters

*color*

The color to show. 1 for Red, 2 for Green, 3 for Blue, and 4 for White.

###### Remarks

### ShutDown Method

#### Shuts down the pattern generator.

##### Syntax

###### Declaration

Protected Friend Overridable Function ShutDown() As Boolean

###### Return Value

True or False; whether or not the PatternGenerator was shut down.

###### Remarks

## Public Properties

### DelayMilliseconds Property

#### Gets or sets the number of milleseconds to wait for the pattern to be rendered on the display.

##### Syntax

###### Declaration

Public Property DelayMilliseconds() As Integer

###### Property Value

The number of milliseconds to wait for a pattern to be rendered on the display

###### Remarks

### HasCustomDeviceForm Property

#### Override and set this property to True if you want to develop a custom form to control the PatternGenerator. The custom form will appear in the View menu of TrueTest.

##### Syntax

###### Declaration

Protected Friend Overridable Property HasCustomDeviceControlForm() As Boolean

###### Property Value

True or False indicating whether there is a custom form

###### Remarks

If you override this property, you should also override the GetCustomForm method.

### IsInitialized Property

#### This property is set to True after the Initialize method is called. It can be overridden for tighter control over the initialization process.

##### Syntax

###### Declaration

Public Overridable Property IsInitialized() As Boolean

###### Property Value

True or False indicating whether the pattern generator is initialized

###### Remarks

If you override this property, you should also override the Initialize method.

# SolidColorPattern Class

## Background

*TrueTestPatternGenerator Namespace*

The SolidColorPattern Class is a simple Pattern which displays a single color across the entire DUT (device under test). SolidColorPattern Class inherits from PatternBase.

## Constructors

### New

#### Initializes a new instance of the SolidColor Class.

##### Syntax

###### Declaration

Public Sub New()

###### Remarks

##### Syntax

###### Declaration

Public Sub New(*Color* as System.Drawing.Color)

**Parameters**

*Color*

The color used to set the SolidColorPattern

###### Remarks

## Public Properties

### PatternColor Property

#### Gets or sets the pattern color.

##### Syntax

###### Declaration

Public Property PatternColor() As System.Drawing.Color

###### Property Value

A color object

###### Remarks

# Common Enumerations – TrueTestEngine Namespace

## TrueTest.APIResult Enum

Whether or not a call to the TrueTest API failed or not.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Success | 0 | API function succeeded. |
| Failed | 1 | API function failed. |
| Canceled | 2 | User canceled the API function. |

## TrueTest.SequenceModeEnum Enum

The sequence mode being used (synchronous or asynchronous).

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Synchronous | 0 | Functions proceed linearly in the order they started. |
| Asynchronous | 1 | Functions can proceed nonlinearly, making use of multithreading and multi-core CPUs. |

## TrueTest.AnalysisResultEnum Enum

Whether or not the result of an analysis (TrueTestEngine.Result) passed or failed.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | The Result was not tested for pass/fail. |
| Pass | 1 | The Result passed. |
| Fail | 2 | The Result failed. |

## TrueTest.OperatingModeEnum Enum

The operating mode, or how TrueTest obtains measurements to analyze.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| UseCamera | 0 | Measurements are obtained from the camera. |
| UseDatabaseAuto | 1 | Measurements are obtained from the database automatically. |
| UseDatabaseUserSelect | 2 | Measurements are obtained from the database by a user selection. |

## TrueTest.AnalysisImageType Enum

The type of measurement being saved or viewed.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| CameraMeasurement | 0 | The original camera measurement. |
| Reference | 1 | A smoothed reference measurement |
| LocalContrast | 2 | A measurement processed to contain Local Contrast. |
| FFT | 3 | A 2D Fourier Transform measurement. |
| Synthetic | 4 | A synthetic measurement, usually built from ROIs. |
| FinalAnalysis | 5 | A processed measurement that includes overlays. |
| RADA | 6 | A rmeasurement of a rectangular device that has been Registered to the device’s active area. |

## TrueTest.CameraRotation Enum

The physical rotation of the camera relative to gravity.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | The camera has not been rotated. |
| Clockwise90 | 90 | The camera has been rotated clockwise by 90 degrees. |
| Rotate180 | 180 | The camera has been rotated 180 degrees. |
| Counterclockwise90 | 270 | The camera has been rotated counterclockwise by 90 degrees. |

# Common Enumerations – RadiantCommon Namespace

## RiBitmapCtl.BlobDrawShape Enum

The shape to use when drawing blobs in the user interface.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| None | 0 | Draws nothing |
| Ellipses | 1 | Draws the best fit ellipse for the blob |
| Contours | 2 | Draws a contour around the blob |
| Blobs | 3 | Draws a contour, but fills in the contour with a new color |

## ColorSpace Enum

The color space type.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| CIE1931xy | 1 | The 1931 x, y color coordinate system |
| CIE1976uv | 2 | The 1976 u’, v’ color coordinate system |
| Tristimulus | 3 | Tristimulus X, Y, Z. |

## DistanceScaleTypeEnum

The distance scale type of a region of interest.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Pixels | 0 | Location in pixels |
| Physical | 1 | Location in physical units. |
| Relative | 2 | Location as a fraction of total. |

## DistanceUnitTypeEnum

The Distance Unit Type for a measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Centimeters | 2 | Centimeters |
| Degrees | 5 | Only valid for intensity measurements. |
| Feet | 3 | Feet |
| Inches | 1 | Inches |
| Meters | 0 | Meters |
| Millimeters | 4 | Millimeters |

## PhotometricTermType Enum

The Photometric Term of the measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| CCT | 13 | Not supported for PM |
| ContrastRatio | 42 | Unit after dividing one measurement from another |
| Gray | 43 | Not supported for PM |
| Illuminance | 0 | An illuminance type measurement |
| Intensity | 2 | An intensity type measurement (created from an illuminance measurement) |
| Lumens | 3 | Not supported for PM |
| Luminance | 1 | A luminance type measurement |
| Millilumens | 4 | Not supported for PM |
| Mura | 40 | Not supported for PM |
| Percent | 41 | Not supported for PM |
| Scatter | 30 | Not supported for PM |

## PhotometricUnitTypeEnum

The photometric unit of a measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Illuminance\_FtCandles | 0 | FootCandles or W/ft2 – Illuminance measurements |
| Illuminance\_KiloLux | 3 | Kilolux or kW/m2 – Illuminance measurements |
| Illuminance\_LumensPerCm | 6 | Lumens/cm2 or W/cm2 – Illuminance measurements |
| Illuminance\_Lux | 1 | Lux or W/m2 – Illuminance measurements |
| Illuminance\_LuxSec | 5 | Lux-Sec or W-s/m2 – Illuminance energy measurements |
| Illuminance\_Megalux | 4 | Megalux or MW/m2 – Illuminance measurements |
| Illuminance\_MilliLux | 2 | Millilux or mW/m2 – Illuminance measurements |
| Intensity\_Candela | 20 | Candela or W/s – Intensity measurements |
| Intensity\_MilliCandela | 21 | Millicandela or mW/s – Intensity measurements |
| Luminance\_CandelaPerCm | 16 | Candela /cm2 or W/sr/cm2 – Luminance measurements |
| Luminance\_CandelaPerMeter | 17 | Candela/m2 or W/sr/m2 – Luminance measurements |
| Luminance\_Footlambert | 10 | Footlambert or W/sr/ft2 – Luminance measurements |
| Luminance\_KiloNit | 13 | Knit or kW/sr/m2 – Luminance measurements |
| Luminance\_MegaNit | 14 | Meganit or MW/sr/m2 – Luminance measurements |
| Luminance\_MilliNit | 12 | Millinit or mW/sr/m2 – Luminance measurements |
| Luminance\_MilliCandelaPerCm | 16 | Millicandela/cm2 or mW/sr/cm2 – Luminance measurements |
| Luminance\_Nit | 11 | Nit or W/sr/m2 – Luminance measurements |

## SpectralResponseTypeEnum

The spectral response of a measurement.

|  |  |  |
| --- | --- | --- |
| **Constant** | **Value** | **Description** |
| Photometric | 1 | Photometric type measurements |
| Radiometric | 0 | Radiometric type measurements |