| **TrueTest CLI API Specification** | | |
|---|---|---|
| Version: | 1.0.1 | |
| Subject: | *CLI API specifications for TrueTest version 1.6.149 or higher.* | Research and Development Application Note |
| Contents: | 17 Pages, No Disks | February 2017 |

# Application Note History

| | |
|---|---|
| Application Note Name: | Park TrueTest CLI API.docx |
| Title: | Park TrueTest CLI API Specifications |
| Product: | TT_API.dll |
| Versions: | MPK_API 1.0.6262.17976 and higher & TT_API 1.0.6194.20809 and higher |
| Original Author: | EMattson |
| Creation Date: | January 2017 |
| Release Date: | February 2017 |

## Updates

| Revision Date | Changes by/ Comments |
|---|---|
| 2017-02-22 | Added 'useLogging' argument to CloseCommunicationAndReinitializeCamera Method to match Initialize Method. Added ability for API to use WhiteListKey file. |
| 2017-03-02 | Updated JSON object key values so user knows how to parse. |
| 2017-05-22 | Updated GetSerialNumber method and Initialization *showFeedbackUI* parameter description. |
| 2017-07-13 | Updated CreateMeasurementSetup method to use additional arguments. |
| 2017-08-17 | Added GetLastMeshData and GetRawData methods to API |
| | |

Radiant Vision Systems, 22908 NE Alder Crest Dr. Ste 100
+1 425 844 0152 Voice • +1 425 844 0153 Fax
Email : support@radiantvs.com
Internet: http://www.radiantvs.com

Radiant Vision Systems, 22908 NE Alder Crest Dr. Ste 100
+1 425 844 0152 Voice • +1 425 844 0153 Fax

Email : support@radiantvs.com
Internet: http://www.radiantvs.com

Radiant Vision Systems, 22908 NE Alder Crest Dr. Ste 100
+1 425 844 0152 Voice • +1 425 844 0153 Fax

Email : support@radiantvs.com
Internet: http://www.radiantvs.com

# Contents

# MPK_API Class - General

## Background

General functions to initialize the camera and retrieve the camera serial number.

## Public Methods

### InitializeCamera Method

*Initializes camera and components of TrueTest.*

#### Syntax

##### Declaration

Public Function InitializeCamera(cameraSerial As String, showFeedbackUI As Boolean, useLogging As Boolean) As JObject

##### Parameters

*cameraSerial*
The camera serial string to initialize.

*showFeedbackUI*
If true, show windows and forms while initializing API / during take measurement. If false this will hide any camera status forms and will also hide lower level message boxes that may appear.

*useLogging*
If true, write to debug log while initializing API.

##### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | InitializationFailure = 2”}

### GetCameraSerial Method

*Returns the attached camera’s serial number.*

#### Syntax

##### Declaration

Public Function GetCameraSerial() As JObject

##### Parameters

*None*

##### Returns

A JSON object. If no camera is attached or powered on, JSON value = “NONE.” If exception occurred, return is JSONUnknownException. {“CameraSerial”: “NONE | *CameraSerialString*”}

If return value is “NONE” then the user should NOT proceed with the camera initialization; otherwise, the camera discovery dialog window will appear.

## EquipmentReady Method

*Returns whether camera has initialized or not.*

### Syntax

#### Declaration

Public Function EquipmentReady() As JObject

#### Parameters

*None*

#### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | InitializationFailure = 2”}

## CloseCommunication Method

*Shuts down API objects and communication to camera.*

### Syntax

#### Declaration

Public Function CloseCommunication() As JObject

#### Parameters

*None*

#### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | InitializationFailure = 2”}

## CloseCommunicationAndReinitializeCamera Method

*Shuts down API objects and communication to camera, then reinitializes connection and communication.*

### Syntax

#### Declaration

Public Function CloseCommunicationAndReinitializeCamera(cameraSerial as String, showFeedbackUI As Boolean, useLogging as Boolean) As JObject

#### Parameters

*cameraSerial*
The camera serial string to initialize.
*showFeedbackUI*
If true, show windows and forms while initializing API / during take measurement.
*useLogging*
If true, write to debug log while initializing API.

#### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | InitializationFailure = 2”}

## GetTrueTestApiVersionInfo Method

*Returns current version of TrueTestEngine DLL.*

### Syntax

#### Declaration

Public Function GetTrueTestApiVersionInfo As JObject

#### Parameters

*none*

#### Returns

A JSON object: {"TrueTestApiVersioninfo":"*VersionInfo"}* or {"ErrorCode": Unknown = 1 | InitializationFailure = 2"}. If success, version returns as File Description + Product Version (major revision.minor version.build number.revision number).

## GetMpkApiVersionInfo Method

*Returns current version of MPK_API DLL.*

### Syntax

#### Declaration

Public Function GetMpkApiVersionInfo As JObject

#### Parameters

*none*

#### Returns

A JSON object: {"MpkApiVersionInfo":"*VersionInfo"}* or {"ErrorCode": Unknown = 1 | InitializationFailure = 2"}. If success, version returns as: File Description + Product Version (major revision.minor version.build number.revision number).

# MPK_API Class - Analysis

## Background

Functions related to preparing and running a pre-defined TrueTest sequence.

## Public Methods

### PrepareForRun Method

*Preps TrueTest to prepare for sequence run.*

#### Syntax

##### Declaration

Public Function PrepareForRun() As JObject

##### Parameters

*None*

##### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1”}

### GetLastResults Method

*Returns the last sequence run results as a list of string.*

#### Syntax

##### Declaration

Public Function GetLastResults() As List(Of String)

##### Parameters

*None*

##### Returns

A List of strings. User must iterate through each result.

### RunAnalysisByName Method

*Runs a specific sequence analysis requiring a single image key.*

#### Syntax

##### Declaration

Public Function RunAnalysisByName(analysisName as String, imageKey as String, xmlParameterSet as String) as JObject

##### Parameters

*analysisName*
The unique name of the analysis to run. This is first defined in the TrueTest software.
*imageKey*
The image key that was defined when taking the measurement.
*xmlParameterSet*
An optional string of xml parameters to override in the current sequence.

#### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | ErrorAnalysisParameter = 6”}

## RunAnalysisByName Method

*Runs a specific sequence analysis requiring an array of image keys.*

### Syntax

#### Declaration

Public Function RunAnalysisByName(analysisName as String, imageKeys() as String, xmlParameterSet as String) as JObject

#### Parameters

*analysisName*
The unique name of the analysis to run. This is first defined in the TrueTest software.
*imageKeys*
The array image keys that was defined when taking the measurement.
*xmlParameterSet*
An optional string of xml parameters to override in the current sequence.

#### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | ErrorAnalysisParameter = 6”}

# MPK_API Class – Data Export

## Background

Functions to export measurement data to disk.

## Public Methods

### ExportData Method

*Exports the luminance and color data to a file for the given image key.*

#### Syntax

##### Declaration

Public Function ExportData(imageHandle as String, path as String, fileName as String) As JObject

##### Parameters

*imageHandle*
The image handle (key) related to the measurement that was captured.
*path*
The location where the exported data is written.
*fileName*
The name of the exported file.

##### Returns

A JSON object: {“ErrorCode”: “Success = 0 | Unknown = 1 | MeasurementNotFound = 4”}

# MPK_API Class – Focus

## Background

Function to understand how well a region of interest is focused.

## Public Methods

### GetFocusMetric Method

*Provides a focus metric for a given region of interest.*

#### Syntax

##### Declaration

Public Function ExportData(imageKey as String, region as Rectangle) As JObject

##### Parameters

*imageKey*
The image handle (key) related to the measurement that was captured.
*region*
The region of interest to calculate the focus metric.

##### Returns

A JSON object: Success = {“FocusMetric”: “*FocusValue*”} / Failure = {“ErrorCode”: “Unknown = 1 | InitializationFailure = 2 | MeasurementNotFound = 4”}

# MPK_API Class – Measurement Setup

## Background

Functions related to setting calibrations and creating a measurement setup used during image capture.

## Public Methods

### CreateMeasurementSetup Method

*Creates a measurement setup which is used during measurement capture.*

#### Syntax

##### Declaration

Public Function CreateMeasurementSetup(description As String, redExposure As Single, greenExposure As Single, blueExposure As Single, xbExposure As Single, focusDistance As Single, lensAperture As Single, autoAdjustExposure As Boolean, subframe As Rectangle, distanceUnit As String, spectralResponseType As String, rotation As Integer) As JObject

##### Parameters

*description*
The name of the measurement setup. This is used when capturing a measurement.
*redExposure*
The desired exposure time for the red filter. Only necessary when making a color measurement.
*greenExposure*
The desired exposure time for the green (photopic) filter. There must always be a value for the green exposure time.
*blueExposure*
The desired exposure time for the blue filter. Only necessary when making a color measurement.
*xbExposure*
The desired exposure time for the xb filter. Only necessary when making a color measurement. Some colorimeters may not be configured with this filter.
*focusDistance*
The physical distance from the DUT at which the camera is focused.
*lensAperture*
The aperture of the camera lens (normally entered as 2.8 or 8.0).
*autoAdjustExposure*
If true, when capturing a measurement, the exposure will be automatically adjusted. This overrides the individual values set in each of the exposure times above.
*subframe*
The region of interest used when capturing a measurement. If the full frame of the ccd is desired, it should be entered as Rectangle(0, 0, totalCcdColumns, totalCcdRows).
*distanceUnit*
The distance unit used when capturing a measurement (entered as “Meters” or “Millimeters”). Only meters or millimeters supported. Default value of meters is used if entered incorrectly.
*spectralResponseType*
The spectral response used when capturing a measurement (entered as “Photometric” or “Radiometric”). Default value of photometric is used if entered incorrectly.

#### Returns

A JSON object: {"ErrorCode": "Success = 0 | Unknown = 1 | InitializationFailure = 2"}

## GetMeasurementSetupNames Method

*Provides a list of the defined measurement setups.*

### Syntax

#### Declaration

Public Function GetMeasurementSetupNames() As List(Of String)

#### Parameters

*None*

#### Returns

A list of strings containing the current measurement setup names. The user must enumerate through each item.

## SetCalibrations Method

*Sets the calibrations used during image capture.*

### Syntax

#### Declaration

Public Function SetCalibrations(measurementSetupName as String, colorCalibrationID as Integer, imageScaleID as Integer, colorShiftID as Integer) As JObject

#### Parameters

*measurementSetupName*
The name of the measurement setup.
*colorCalibrationID*
The ID of the color calibration. Use Function GetColorCalibrationList() to find out possible IDs to set.
*imageScaleID*
The ID of the image scale calibration. Use Function GetImageScaleCalibrationList() to find out possible IDs to set.
*colorShiftID*
The ID of the color shift calibration. Use Function GetColorShiftCalibrationList() to find out possible IDs to set.

#### Returns

A JSON object: {"ErrorCode": "Success = 0 | Unknown = 1 | MeasurementSetupNotFound = 5"}

## GetColorCalibrationList Method

*Returns the color calibrations.*

### Syntax

#### Declaration

Public Function GetColorCalibrationList() As JObject

#### Parameters

*None*

#### Returns

A JSON object showing key, value pair of calibration name and ID. If success, error code: 0. If exception occurred, JSONUnknownException (error code 1).

## GetImageScaleCalibrationList Method

*Returns the image scale calibrations.*

### Syntax

#### Declaration

Public Function GetImageScaleCalibrationList() As JObject

#### Parameters

*None*

#### Returns

A JSON object showing key, value pair of calibration name and ID. If success, error code: 0. If exception occurred, JSONUnknownException (error code 1).

## GetColorShiftCalibrationList Method

*Returns the image scale calibrations.*

### Syntax

#### Declaration

Public Function GetColorShiftCalibrationList() As JObject

#### Parameters

*None*

#### Returns

A JSON object showing key, value pair of calibration name and ID. If success, error code: 0. If exception occurred, JSONUnknownException (error code 1).

## FlushMeasurementSetups Method

*Clears all defined measurement setups from memory.*

### Syntax

#### Declaration

Public Function FlushMeasurementSetups() As JObject

#### Parameters

*None*

#### Returns

A JSON object {“ErrorCode”:”Success = 0”}

# MPK_API Class – Measurement

## Background

Functions related to capturing measurements.

## Public Methods

### CaptureMeasurement Method

*Captures a measurement using the pre-defined measurement setup and stores it in memory using the image key.*

#### Syntax

##### Declaration

Public Function CaptureMeasurement(measurementSetupName as String, imageKey as String, saveToDatabase as Boolean) As JObject

##### Parameters

*measurementSetupName*
The name of the measurement setup.
*imageKey*
The key used to retrieve the measurement from memory.
*saveToDatabase*
If true, the measurement will be saved in the TrueTest database.

##### Returns

A JSON object: Success = {“Imagekey”:”*Imagekey*”} / Failure = {“ErrorCode”: “Unknown = 1 | ErrorMeasurement = 3”}

### GetMeasurementNames Method

*Get the current image key names stored in memory.*

#### Syntax

##### Declaration

Public Function GetMeasurementNames() As List(Of String)

##### Parameters

*None*

##### Returns

A list of strings of the measurement (image key) names. User must enumerate through each item.

### FlushMeasurements Method

*Clears the captured measurements from memory.*

#### Syntax

##### Declaration

Public Function FlushMeasurements() As JObject

##### Parameters

*None*

### Returns

A JSON object {“ErrorCode”:”Success = 0“}

Radiant Vision Systems, 22908 NE Alder Crest Dr. Ste 100
+1 425 844 0152 Voice • +1 425 844 0153 Fax

Email : support@radiantvs.com
Internet: http://www.radiantvs.com

# MPK_API Class – Measurement Data

## Background

Functions related to directly retrieving measurement data.

## Public Methods

### GetLastMeshData Method

*Retrieves the mesh (synthetic image) data from the analysis that was last executed.*

#### Syntax

##### Declaration

Public Function GetLastMeshData() as List(Of Single(,))

##### Parameters

*None*

##### Returns

A list of 2D arrays containing the mesh data from the analysis that was last executed. The user must enumerate through each item in the list. If the measurement was monochromatic, the list will only contain a single array for brightness, otherwise it will contain arrays for color as well. Arrays contained in the list are in the following order: Lv, Cx, Cy, u', v'.

### GetRawData Method

*Retrieves the raw data from the camera measurement.*

#### Syntax

##### Declaration

Public Function GetRawData(imageKey as String) as List(Of Single(,))

##### Parameters

*imageKey*

The key used to retrieve the measurement from memory.

##### Returns

A list of 2D arrays containing the raw data from the measurement retrieved via the image key. The user must enumerate through each item in the list. If the measurement was monochromatic, the list will only contain a single array for brightness, otherwise it will contain arrays for color as well. Arrays contained in the list are in the following order: Lv, Cx, Cy, u', v'.

# Appendix I

## White List Key file

The ability to use a white list key file – to allow TrueTest to run without a dongle attached – has been added for MPK_API version > 1.0.6262.17976 and TrueTestEngine version > 1.2.0.1136.

The WhiteListSecurity.xml file must be in the \Radiant Vision Systems Data\TrueTest\AppData folder.

As of this current writing, February 23, 2017, Radiant Imaging Colorimeters and Photometers still require the entry of the PMEngine license code *if and only if*:

1. The PMEngine license code hasn't been previously entered for a particular camera purchased before 2017/02/23.
2. If the camera calibration database (.calx) for a camera, that was used on a system and that contains the entered PMEngine license code, is not moved to the new system where the camera is being used.
    a. If the calibration file is new, meaning it was copied from the camera, and the camera was received before the date listed above, a message box will appear requiring the user to enter the PMEngine license code.
    b. To avoid the PMEngine license code pop-up, the user shall copy the previously used calx camera calibration file to the new machine, or use a local (network) location that houses all calx camera calibration files.

For cameras purchased *after* the listed date above: the user will not be required to enter a PMEngine license code.