

## ***Software Development Kit***

# **IQA**

*Copyright 2019*

***Recogniform Technologies SpA***

# HOW TO CONTACT US

Recogniform Technologies SpA  
Contrada Concistocchi  
87036 Rende (CS), Italy

Phone : +39 0984 404174

Fax : +39 0984 830299

Internet : [www.recogniform.com](http://www.recogniform.com)

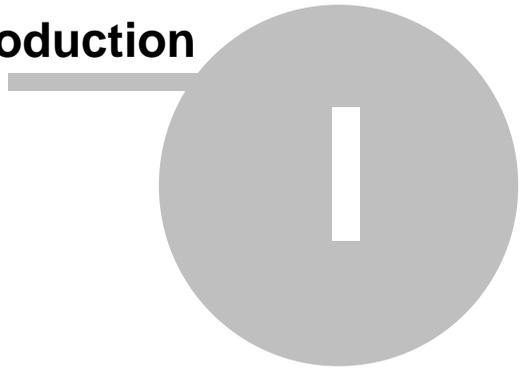
E-Mail : [info@recogniform.com](mailto:info@recogniform.com)

# Table of contents

<b>Introduction</b>	<b>6</b>
Copyright .....	6
License .....	6
Overview .....	6
<b>Usage</b>	<b>24</b>
Visual C++ .....	24
C# .....	24
Delphi .....	24
Visual Basic .....	24
Visual Basic .NET .....	24
Java .....	24
<b>Used data types</b>	<b>26</b>
<b>API References</b>	<b>28</b>
IQA_Init .....	28
IQA_Done .....	29
IQA_Execute .....	30
IQA_SetParameter .....	31
IQA_GetParameter .....	34
IQA_GetMeasure .....	36
IQA_GetMeasureName .....	39
LoadIQALibrary .....	42
FreeIQALibrary .....	42
<b>Samples</b>	<b>46</b>
Visual C++ .....	46



# Introduction



# 1 Introduction

## 1.1 Copyright

The software and the documentation are property of:

Recogniform Technologies SpA  
Contrada Concistocchi  
87036 Rende (CS)  
Italy  
www.recogniform.com  
info@recogniform.com

## 1.2 License

It is illegal to copy or reproduce this manual, or any part thereof, in any shape or form.

The information contained in this manual is subject to change without notice and does not present a commitment on the part of Recogniform Technologies SpA.

Recogniform Technologies SpA shall not be held liable for technical or editorial errors and/or omissions made here, nor for incidental or consequential damages resulting from the furnishing, performance, or use of the software and documentation.

Recogniform Technologies SpA reserves the right to make changes to the software and documentation without notice.

Product names mentioned here are used for identification purposes only and may be tradenames and/or registered trademarks of their respective companies.

YOU CANNOT DISTRIBUTE SOFTWARE INCLUDING THIS SDK LIBRARY UNLESS YOU HAVE A WRITTEN AGREEMENT (ROYALTIES FREE OR ROYALTIES BASED) WITH RECOGNIFORM TECHNOLOGIES SPA

## 1.3 Overview

The Recogniform IQA Library 2.1 allows to perform image quality assurance according to Check21 requirements.

The system performs quantitative measures of images attributes (e.g. "image skew") and a qualitative assertion about the presence

of a defect (e.g. "excessive document skew") using user-defined threshold parameters (e.g. "minimum and maximum skew angle allowed") that strike an effective and practical balance between correctly identifying defects that might affect usability (avoiding "escapes") and incorrectly identifying defects that won't affect usability ("false positives").

Once the defect is determined, it will be possible to "repair" the defect without re-scan using Image Processing technique (deskew, despeckle, black border removal, dynamic thresholding, etc.), or by a new scanning action: this last option is strictly necessary when the image lost a portion of its content, as caused by improper positioning/ feeding on scanner device.

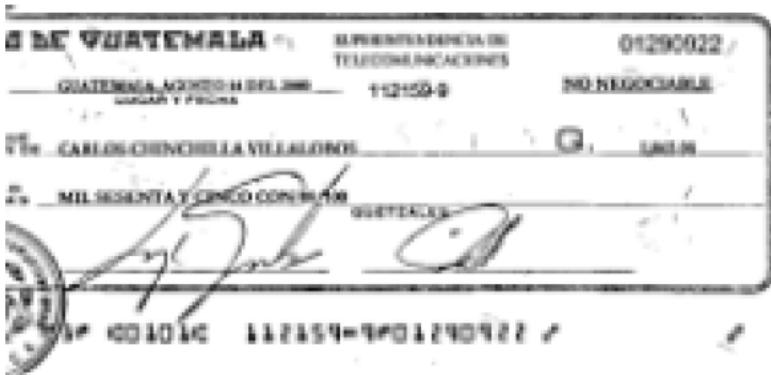
Some of possible business impacts of image quality defects are:

- Missing information due to missing/obscured key data fields.
- Financial losses due to document re-scan (work flow and labor expense impact).
- Negative impact on optical recognition
- Esthetic artifacts with the document image that could create issues for image print applications.
- Legibility and usability problems.
- Missing image/document during the image capture process due to under-spaced documents.
- No information on the piggybacked item (item behind).
- Potential losses due to lost information.

In next sections you can find an overview of each image defects detected.

### **Undersize Image**

This defect is due to the document image rendition's width or height being below the minimum image size based on the minimum image size and tolerances associated with the image capture platform.



Condition:

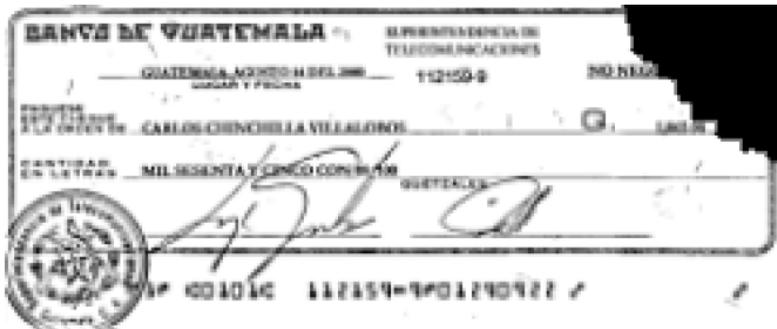
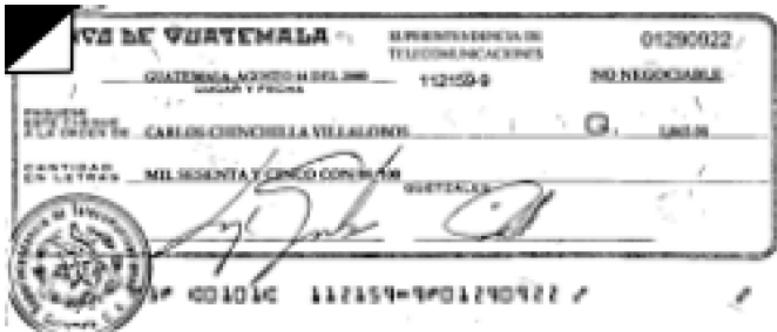
Image width/height < minimum user-defined width/height threshold.

This defect can develop in case of:

- torn document (a significant portion of the original source document is absent);
- folded document (a significant portion of the original source document is folded);
- improperly framed document (the leading or trailing edge of the document has been truncated due to a camera synchronization error during the image capture process).

### **Folded or Torn Document Corners**

This defect is due to the corner of the source document being either missing and/or folded in the document image rendition.



Condition:  
 Fold/tear corner width/height > Maximum fold/tear corner  
 width/height threshold.

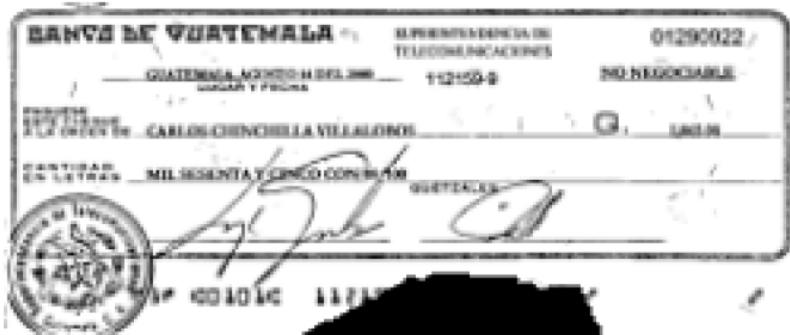
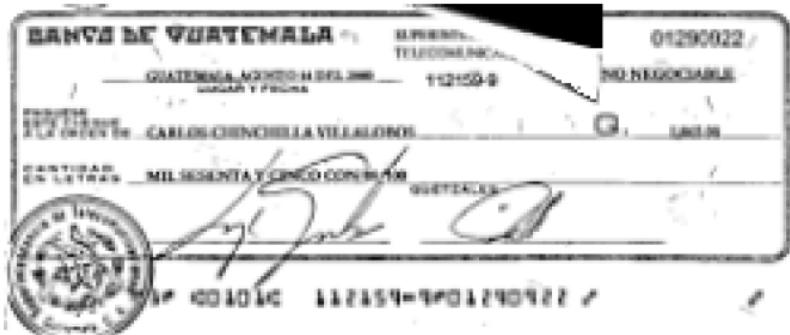
Independent fold/tear corner width and height thresholds will be defined for each corner of the document, i.e., four separate sets of width and height thresholds.

This defect can develop in case of:

- Folded document corners (a corner of the source document has been folded, causing an area of the document image to be missing and obscured).
- Torn document corners (a missing corner in the source document, resulting in an area of the document image to be missing)

### Folded or Torn Document Edges

This defect is due to the edge of the source document being either missing and/or folded in the document image rendition.



Condition:

Folded/torn edge width > Maximum edge fold/tear width threshold

and

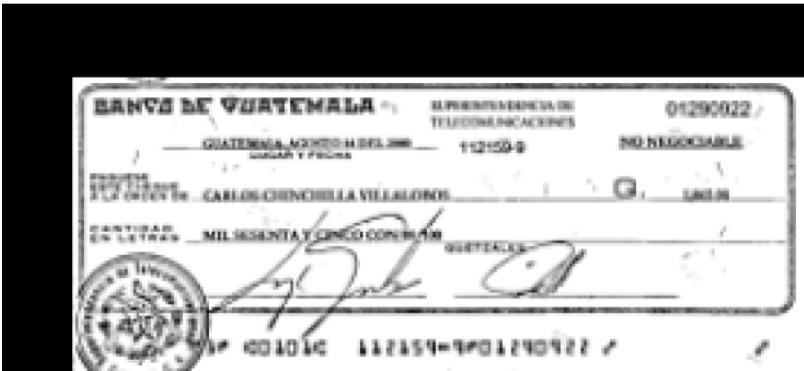
Folded/torn edge height > Maximum edge fold/tear height threshold

Independent folded/torn edge width and height thresholds will be defined for each edge of the document, i.e., four separate sets of width and height thresholds.

This defect can develop in case of torn and/or folded document edge (an edge of the source document has been torn and/or folded, causing an area of the document image to be missing and obscured).

### Document Framing Error

This defect is due to the inclusion of additional vertical and/or horizontal scan lines, within the document image, that contain no document pixel data.



Condition:

Width of additional left/right/top/bottom edge scan lines > Maximum left/right/top/bottom edge over scan threshold.

This defect can develop in case of:

- Presence of additional scan lines prior to the left (or prior to the right) edge of the document in the document image.
- Presence of additional scan lines below the bottom (or above the top) edge of the document in the document image.

These conditions can be the result of the image camera system not being able to properly detect the edges of the document during the image capture process.

### Excessive Document Skew

This defect is caused by the image document not being in proper alignment with the image camera sensor.



Condition:

Document Skew Angle < Negative Skew Angle Threshold

or

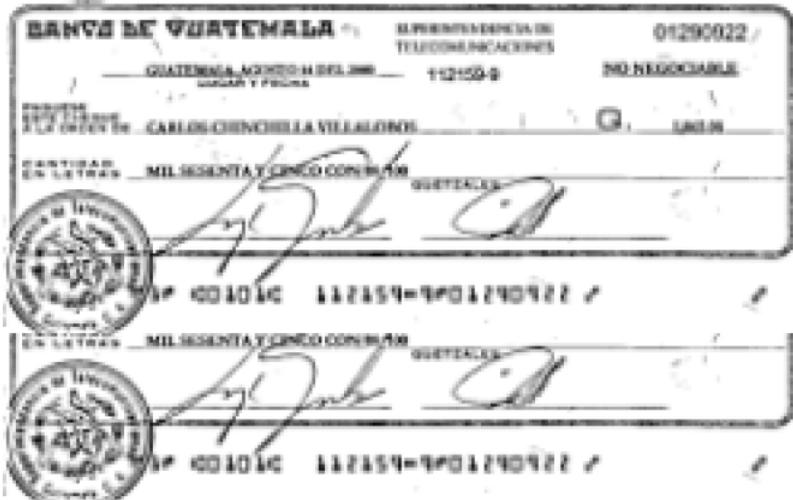
Document Skew Angle > Positive Skew Angle Threshold

This defect can appear in case of:

- Paper handling problems in the document transport, e.g. document feeder, transport belts/rollers. Documents may not be properly aligned in the transport track, resulting in the document being skewed as it imaged by the camera subsystem.
- Improper alignment of the document on a flatbed scanner. If the document is being imaged using a flatbed scanner, improper alignment of the document on the scanner window will result in a skewed document image.

### Oversize Image

This defect is due to the document image rendition's width or height being above the maximum image size based on a maximum image size and tolerances associated with the image capture platform.



Condition:

Image width > Maximum image width threshold

or

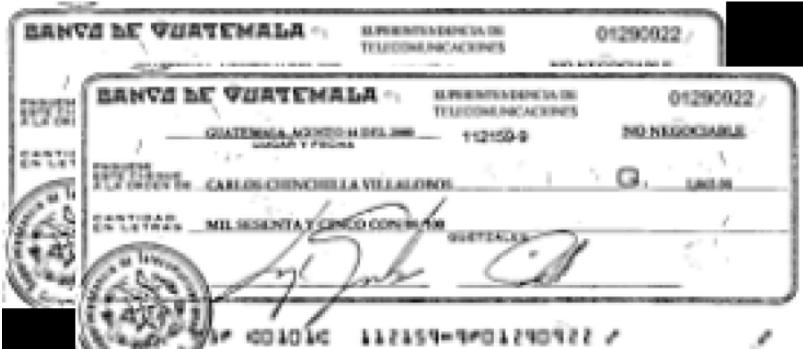
Image height > Maximum image height threshold

It may verify in case of:

- Overlapped (piggy-backed) documents. An image containing two or more documents that are overlapped as they pass the image camera
- Under-spaced documents. An image containing two or more documents that are separated by only a small distance (or end-to-end), resulting in two documents being captured as a single image.
- Skewed documents. Excessively skewed documents may cause the maximum image height to be exceeded

### Piggyback Document

A piggy-back image defect occurs when two or more documents are present and overlapped within the document image.



It may be identified by:

- Document images with more than one document.
- Detected multiple document heights within the document image.

This defect is generally due to:

- Poor document quality.
- Poor document work preparation/sorting.
- Mechanical handling and control problems within the document transport feeder or track

### Image Too Light

For a bi-tonal image, the defect is due to a insufficient number of "black" pixels. For a gray level image, the defect is due to high "brightness" and low "contrast".



Condition for Bi-tonal Images:

Percentage of black pixels < Minimum percentage of black pixels threshold

Condition for Gray Level and Color Images:

% Average Brightness > Maximum percent brightness threshold

and

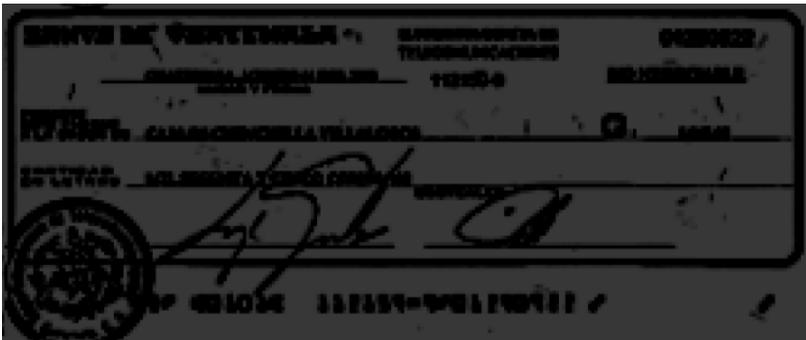
% Average Contrast < Minimum percent contrast threshold

This defect can be due to one or more of the following problems:

- Poor printing/writing contrast on the source document.
- Improper thresholding of the document background.
- Illumination problems with the image capture subsystem.
- Image camera calibration problems.

### **Image Too Dark**

For a bi-tonal image, this defect is due to the image having too many "black" pixels. For a gray level image, this defect is due to the image having insufficient "brightness".



Condition for Bi-tonal Images:

% of black pixels > Maximum % of Black Pixels Threshold.

Condition for Gray Level and Color Images:

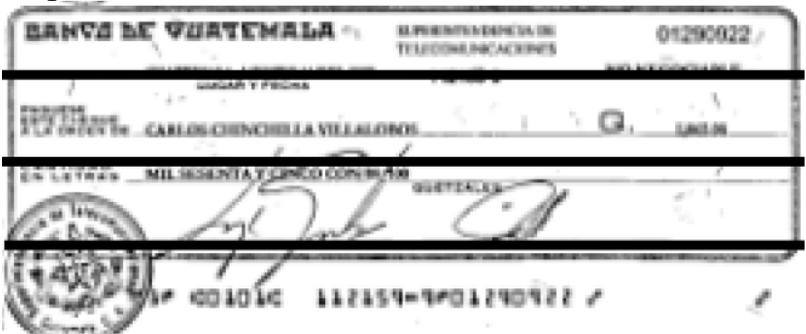
% Average Brightness < Minimum % Brightness Threshold

This defect could be an indicator of one of the following problems:

- Excessive printing/writing on the source document.
- Improper thresholding of the document background.
- Large amounts of black pixel "noise" present in the image.
- Illumination problems with the image capture subsystem.
- Image camera calibration problems.

### Horizontal Streaks Present in the Image

A defect due to the image containing one or more "dark" (for all images) or "light" (for gray level and color images) horizontal streaks that extend horizontally across the majority of the entire document image.



Condition for Bi-tonal Images:

Largest black streak height > Maximum Black Streak Height Threshold

Number of black streaks > Maximum Black Streak Count Threshold

Condition for Gray Level and Color Images

Largest gray level streak height > Maximum Gray Level Streak Height Threshold

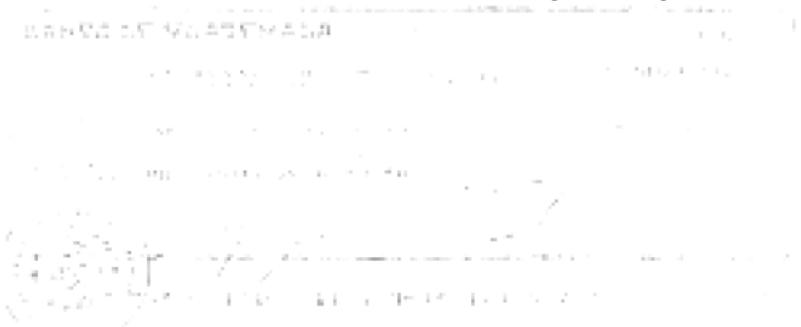
Number of gray level streaks > Maximum Gray Level Streak Count Threshold

Dark streaks can be caused by a number of factors during the image capture process. Possible sources of dark streaks include the following:

- Dirt and/or ink that can adhere to either the image capture scan window or camera lens commonly present in most high, medium or low-speed document transport imaging systems.
- A scratch or irregularity present on the image scan window or camera lens top or bottom.
- Dirt or debris on camera calibration targets, i.e., white reference targets.
- Failure of the image camera CCD sensor or electronics.

### **Below Minimum Compressed Image Size**

A defect due to the invalid size of file containing the image.



Condition for bi-tonal images:

Compressed Image Size < Minimum Bi-tonal Compressed Image Size Threshold

Condition for gray level and color images:

Compressed Image Size < Minimum Gray Level Compressed Image Size Threshold

Minimum compressed image size thresholds will be independently established for the front and rear image of the document and will be dependent on the image compression method utilized.

This defect could be an indicator of one of the following problems:

- Improper suppression (thresholding) of the document background.
- Image camera calibration problems.
- Inappropriate compression parameters/settings, yielding an image with a high level of distortion.
- A white document with very little writing or printing, e.g., the rear of a image with a small endorsement.

### **Above Maximum Compressed Image Size**

A defect due to the invalid size of file containing the image.



Condition for bi-tonal images:

Compressed Image Size > Maximum Bi-tonal Compressed Image Size Threshold

Condition for gray level and color images:

Compressed Image Size > Maximum Gray Level Compressed Image Size Threshold

Maximum compressed image size thresholds will be independently established for the front and rear image of the document and will be dependent on the image compression method utilized.

A large compressed image packet size is generally an indicator of an image with a high information content, e.g., lots of writing or printing or high contrast background patterns. In the case of a bi-tonal image rendition, a large compressed image packet size occurs when the image contains a lot of black/white pixel transitions. This could be an indicator that the bi-tonal image has the following attributes:

- a significant amount of image "noise" present in the image;
- a large amount of written/printed data present in the image;
- a significant amount of the image background pattern/scene has

- been retained during the creation of the bi-tonal rendition.

### Excessive "Spot Noise" in the Image

Spot noise is an image defect due to image containing "excessive occurrences" (greater than some defined count) of "spot noise" (isolated dark small pixel groups).



Condition:

Average spot noise count per 1 sq.inch area > Maximum spot noise count threshold.

Noise can be caused by one or more of factors:

- The scanned document may have a "cluttered" background such as a complex high contrast image. When imaged and thresholded, this type of background can result in many small dark regions or noise.
- Noise can result from a document that has very low contrast. In this case the threshold algorithm may produce many isolated dark regions as it struggles to differentiate between what is dark and what is bright. This can happen if the original gray-scale image is bright or dark.
- Low contrast and subsequent noise can also occur if there is a problem with the scanning system such as improper illumination.
- Noise could be the result of physical defects on the document being scanned.
- The surface of an item may contain actual dark regions resulting from dirt or other contaminants that will result in a noisy image

### Front-Rear Image Dimension Mismatch

This type of defect indicates that the image height and width do not match between the front and rear images of the source document.



Condition:

Absolute value of the image width difference > Maximum image width difference

or

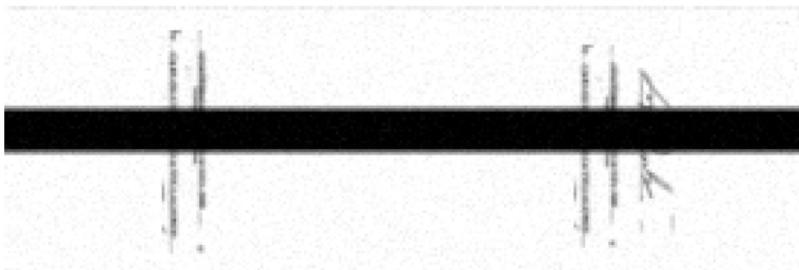
Absolute value of the image height difference > Maximum image height difference

This defect may verify in case of:

- The front image of document "n" being matched up with the rear image of document "n-1".
- Differences in document framing for the front and rear image renditions of the document.

### **Carbon Strip Detected**

A defect due to the presence of a "carbonized band", that typically extends from the leading edge to trailing edge on the rear of the image, that can potentially interfere with the legibility of endorsements.

**Condition:**

A black band is present on the rear of the document image that meets the size and location requirements for a carbonized band. The black band height exceeds the Minimum Carbon Strip Height parameter setting.

This defect is generally due to:

- Presence of a carbon strip printed on the rear of the image that facilitates the transfer of information from the image to another document.
- Continued use of carbon has been driven by customer preference for the higher transfer capability and readability it provides.

**Image "Out of Focus"**

An image defect due to the image camera subsystem being "out of focus" resulting in blurred image renditions of the document.

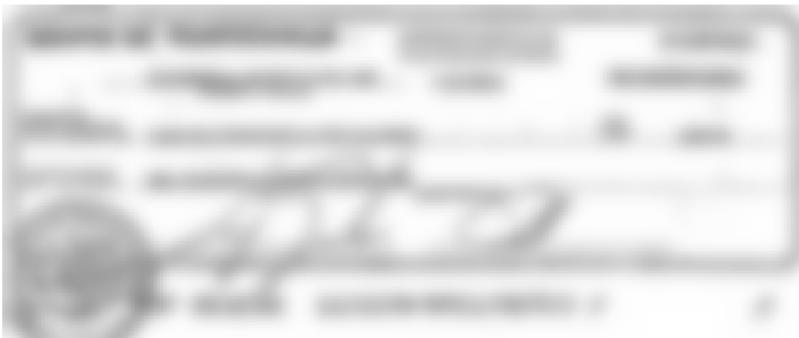
**Condition:**

Image Focus Score < Minimum Focus Threshold

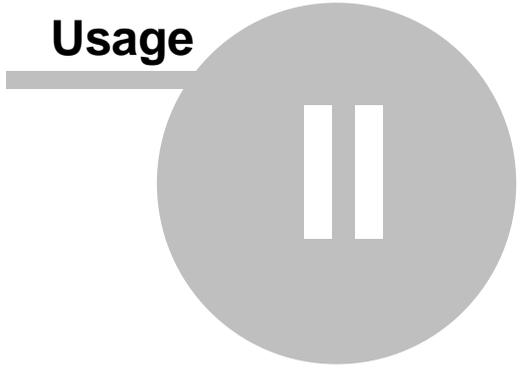
This defect is generally due to:

- A change in the image camera's optical-mechanical settings.
- Imaging of a document that is not positioned within the image

camera's "depth of focus"



**Usage**



## 2 Usage

### 2.1 Visual C++

You have to include the RECOIQAAPI.C in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .exe directory or in windows\system directory.

### 2.2 C#

You have to include the RECOIQAAPI.CS in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .exe directory or in windows\system directory.

### 2.3 Delphi

You have to include the RECOIQAAPI.PAS in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .exe directory or in windows\system directory.

### 2.4 Visual Basic

You have to include the RECOIQAAPI.BAS in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .exe directory or in windows\system directory.

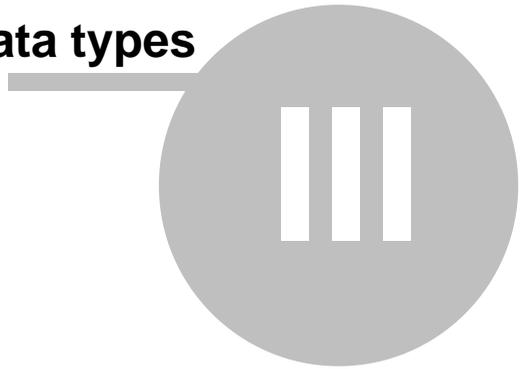
### 2.5 Visual Basic .NET

You have to include the RECOIQAAPI.VB in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .exe directory or in windows\system directory.

### 2.6 Java

You have to use 32 bit JVM and you have to include the INTERFACEIQA.java in your program. Before to execute your application make sure the *RECOIQA.DLL* is available in your same .jar directory.

**Used data types**



### **3 Used data types**

All the numeric parameters passed and returned by the DLL functions are 32 bit integers.

All the string parameters passed and returned by the DLL functions are ANSI null terminated strings.

All the functions have to be called with "stdcall" convention.

All the images handles have to be memory handles (allocated using GlobalAlloc Windows function) packed as DIB (Device Independent Bitmap) format following Microsoft specifications.

## **API References**



**IV**

## 4 API References

### 4.1 IQA\_Init

#### C/C++ Declaration

```
long IQA_Init(char *User, char *Password);
```

#### C# Declaration

```
int IQA_Init(string Name, string Key);
```

#### Visual Basic Declaration

```
Function IQA_Init (ByVal Name As String, ByVal Key  
As String) As Long
```

#### Visual Basic .NET Declaration

```
Function IQA_Init(ByVal Name As String, ByVal Key  
As String) As Integer
```

#### Delphi Declaration

```
function  
IQA_Init(Company,LicenseKey:PAnsiChar):Integer;  
stdcall;
```

#### Java Declaration

```
int IQA_Init(String User, String Password);
```

#### Description

This is the first function to call: initialize the library and returns a session handle to use in next calls. When you buy the library you receive an "user" and a "password" string necessary to initialize the library in normal mode: without this value or with wrong values the library is initialized in evaluation mode. The evaluation mode works exactly as normal mode but some time, when you call a function, is displayed a warning dialog box remembering the evaluation state: you can close it and continue to work with no problems.

#### Parameters

*User* (in) - then user name string  
*Password* (in) - then password string

### Return values

The session handle if the library is initialized, 0 otherwise.

## 4.2 **IQA\_Done**

### **C/C++ Declaration**

```
void IQA_Done(long SessionHandle);
```

### **C# Declaration**

```
void IQA_Done(int Session);
```

### **Visual Basic Declaration**

```
Sub IQA_Done (ByVal Session As Long)
```

### **Visual Basic .NET Declaration**

```
Sub IQA_Done(ByVal Session As Integer)
```

### **Delphi Declaration**

```
procedure IQA_Done(SessionHandle:Integer);  
stdcall;
```

### **Java Declaration**

```
void IQA_Done(int SessionHandle);
```

### **Description**

This is the last function to call when you don't need more services from the library: deinitialize the library and free all used resources.

### **Parameters**

*SessionHandle* (in) - the session handle to free

### **Return values**

n/a

## 4.3 IQA\_Execute

### C/C++ Declaration

```
RECOIQA_API long IQA_Execute(long SessionHandle,  
long FrontDIBHandle, long BackDIBHandle, char  
*FileName);
```

### C# Declaration

```
int IQA_Execute(int Session, int FrontDIBHandle,  
int BackDIBHandle, string FileName);
```

### Visual Basic Declaration

```
Function IQA_Execute (ByVal Session As Long, ByVal  
FrontDIBHandle As Long, ByVal BackDIBHandle As  
Long, ByVal FileName As String) As Long
```

### Visual Basic .NET Declaration

```
Function IQA_Execute(ByVal Session As Integer,  
ByVal FrontDIBHandle As Integer, ByVal  
BackDIBHandle As Integer, ByVal FileName As  
String) As Integer
```

### Delphi Declaration

```
function  
IQA_Execute(SessionHandle:Integer;FrontDIBHandle,B  
ackDIBHandle:Integer; FileName:PAnsiChar):Integer;  
stdcall;
```

### Java Declaration

```
int IQA_Execute(int SessionHandle, int  
FrontDIBHandle, int BackDIBHandle,String  
FileName);
```

### Description

This is the function performing the image quality analysis. You have to call this function before to use [IQA\\_GetMeasure](#) function to retrieve IQA flags or measures representing the image quality.

### Parameters

*SessionHandle* (in) - the session handle to use. Its the value returned by library initialization function [IQA\\_Init](#).

*FrontDIBHandle* (in) - the handle of the DIB with the front side image of the check/image to process.

*BackDIBHandle* (in) - the handle of the DIB with the back side image of the check/image to process. If you want to process just front side you can use the "0" value instead that a valid memory handle, but consider that some checking cannot be performed without the back image.

*FileName*(in) - the complete file name path where the file name with front and back images exists. If you haven't a file name you can use the NULL value instead that a valid file name, but consider that some checking cannot be performed without a valid file name.

#### Return values

0 if success, 0xFFFFFFFF if some error occurred.

#### Note

You can fine tune the process to execute setting properly the parameters with [IQA\\_SetParameter](#) before to call this function.

## 4.4 **IQA\_SetParameter**

### C/C++ Declaration

```
RECOIQA_API long IQA_SetParameter(long  
SessionHandle, long Parameter, long Value);
```

### C# Declaration

```
int IQA_SetParameter(int Session, int Parameter,  
int Value);
```

### Visual Basic Declaration

```
Function IQA_SetParameter (ByVal Session As Long,  
ByVal Parameter As Long, ByVal Value As Long) As  
Long
```

### Visual Basic .NET Declaration

```
Function IQA_SetParameter(ByVal Session As  
Integer, ByVal Parameter As Integer, ByVal Value  
As Integer) As Integer
```

## Delphi Declaration

```
function  
IQA_SetParameter(SessionHandle: Integer; Parameter: Integer; Value: Integer): Integer; stdcall;
```

## Java Declaration

```
int IQA_SetParameter(int SessionHandle, int  
Parameter, int Value);
```

## Description

This function set a custom parameter into the current IQA session.

## Parameters

*SessionHandle (in)* - the session handle to use. Its the value returned by library initialization function [IQA\\_Init](#).

*Parameter (in)*: the parameter to set. Allowed values are:

- 0: Left Margin to skip performing QC in 1/100 of inch (default is 25)
- 1: Top Margin to skip performing QC in 1/100 of inch (default is 25)
- 2: Right Margin to skip performing QC in 1/100 of inch (default is 25)
- 3: Bottom Margin to skip performing QC in 1/100 of inch (default is 25)
- 4: Minimum Image Size Threshold Horizontal to pass QC in 1/100 of inch (default is 0)
- 5: Minimum Image Size Threshold Vertical to pass the QC in 1/100 of inch (default is 0)
- 6: Maximum Image Size Threshold Horizontal to pass the QC in 1/100 of inch (default is maxint)
- 7: Maximum Image Size Threshold Vertical to pass the QC in 1/100 of inch (default is maxint)
- 8: Maximum Left Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
- 9: Maximum Right Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
- 10: Maximum Top Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
- 11: Maximum Bottom Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
- 12: Maximum Top Left Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)

- 
- 13: Maximum Top Right Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 14: Maximum Bottom Left Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 15: Maximum Bottom Right Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 16: Maximum Top Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 17: Maximum Bottom Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 18: Maximum Right Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 19: Maximum Left Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 20: Max Negative Skew Angle Threshold to pass QC in 1/10 of degree (default is  $-5 = -0.5^\circ$ )
  - 21: Max Positive Skew Angle Threshold to pass QC in 1/10 of degree (default is  $5 = 0.5^\circ$ )
  - 22: Minimum Percentage Black Pixel Threshold to pass QC in 1/10 of percentage (default is 30 = 3%)
  - 23: Maximum Percentage Black Pixel Threshold to pass QC in 1/10 of percentage (default is 200 = 20%)
  - 24: Minimum Percent Brightness Threshold to pass QC in 1/10 of percentage (default is 700 = 70%)
  - 25: Maximum Percent Brightness Threshold to pass QC in 1/10 of percentage (default is 900 = 90%)
  - 26: Minimum Percent Contrast Threshold to pass QC in 1/10 of percentage (default is 500 = 50%)
  - 27: Maximum Spot Noise Count Threshold to pass QC in pixels per square inch (default is 50)
  - 28: Minimum Focus Threshold to pass QC in 1/10 of percentage (default is 500 = 50%)
  - 29: Minimum Bitonal Compressed Image Size Threshold to pass QC in bytes (default is 10000)
  - 30: Maximum Bitonal Compressed Image Size Threshold to pass QC in bytes (default is 100000)
  - 31: Minimum Graylevel Compressed ImageSize Threshold to pass QC in bytes (default is 50000)
  - 32: Maximum Graylevel Compressed Image Size Threshold to pass QC in bytes (default is 500000)
  - 33: Maximum Image Width Difference with back side to pass QC in 1/100 of inch (default is 8)
  - 34: Maximum Image Height Difference with back side to pass QC in 1/100 of inch (default is 8)

*Value:* the value to set as current parameter. See Parameter to see allowed values.

### Return values

0 if success, 0xFFFFFFFF if some error occurred.

## 4.5 **IQA\_GetParameter**

### **C/C++ Declaration**

```
RECOIQA_API long IQA_GetParameter(long  
SessionHandle, long Parameter);
```

### **C# Declaration**

```
int IQA_GetParameter(int Session, int Parameter);
```

### **Visual Basic Declaration**

```
Function IQA_GetParameter (ByVal Session As Long,  
ByVal Parameter As Long) As Long
```

### **Visual Basic .NET Declaration**

```
Function IQA_GetParameter(ByVal Session As  
Integer, ByVal Parameter As Integer) As Integer
```

### **Delphi Declaration**

```
function  
IQA_GetParameter(SessionHandle:Integer;Parameter:I  
nteger):Integer; stdcall;
```

### **Java Declaration**

```
int IQA_GetParameter(int SessionHandle, int  
Parameter);
```

### **Description**

This function get a custom parameter into the current IQA session.

### **Parameters**

*SessionHandle (in)* - the session handle to use. Its the value returned by library initialization function [IQA\\_Init](#).

*Parameter (in)*: the index of parameter to get. Allowed values are:

0: Left Margin to skip performing QC in 1/100 of inch (default is

- 
- 25)
  - 1: Top Margin to skip performing QC in 1/100 of inch (default is 25)
  - 2: Right Margin to skip performing QC in 1/100 of inch (default is 25)
  - 3: Bottom Margin to skip performing QC in 1/100 of inch (default is 25)
  - 4: Minimum Image Size Threshold Horizontal to pass QC in 1/100 of inch (default is 0)
  - 5: Minimum Image Size Threshold Vertical to pass the QC in 1/100 of inch (default is 0)
  - 6: Maximum Image Size Threshold Horizontal to pass the QC in 1/100 of inch (default is maxint)
  - 7: Maximum Image Size Threshold Vertical to pass the QC in 1/100 of inch (default is maxint)
  - 8: Maximum Left Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
  - 9: Maximum Right Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
  - 10: Maximum Top Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
  - 11: Maximum Bottom Edge Overscan Threshold to pass the QC in 1/100 of inch (default is 0)
  - 12: Maximum Top Left Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 13: Maximum Top Right Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 14: Maximum Bottom Left Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 15: Maximum Bottom Right Corner Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 16: Maximum Top Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 17: Maximum Bottom Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 18: Maximum Right Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 19: Maximum Left Border Fold Threshold to pass QC in 1/100 of inch (default is 3)
  - 20: Max Negative Skew Angle Threshold to pass QC in 1/10 of degree (default is  $-5 = -0.5^\circ$ )
  - 21: Max Positive Skew Angle Threshold to pass QC in 1/10 of degree (default is  $5 = 0.5^\circ$ )
  - 22: Minimum Percentage Black Pixel Threshold to pass QC in 1/10 of percentage (default is  $30 = 3\%$ )
  - 23: Maximum Percentage Black Pixel Threshold to pass QC in

1/10 of percentage (default is 200 = 20%)  
24: Minimum Percent Brighness Threshold to pass QC in 1/10 of percentage (default is 700 = 70%)  
25: Maximum Percent Brighness Threshold to pass QC in 1/10 of percentage (default is 900 = 90%)  
26: Minimum Percent Contrast Threshold to pass QC in 1/10 of percentage (default is 500 = 50%)  
27: Maximum Spot Noise Count Threshold to pass QC in pixels per square inch (default is 50)  
28: Minimum Focus Threshold to pass QC in 1/10 of percentage (default is 500 = 50%)  
29: Minimum Bitonal Compressed Image Size Threshold to pass QC in bytes (default is 10000)  
30: Maximum Bitonal Compressed Image Size Threshold to pass QC in bytes (default is 100000)  
31: Minimum Graylevel Compressed ImageSize Threshold to pass QC in bytes (default is 50000)  
32: Maximum Graylevel Compressed Image Size Threshold to pass QC in bytes (default is 500000)  
33: Maximum Image Width Difference with back side to pass QC in 1/100 of inch (default is 8)  
34: Maximum Image Height Difference with back side to pass QC in 1/100 of inch (default is 8)

### Return values

The current setting or 0xFFFFFFFF if some error occurred.

## 4.6 IQA\_GetMeasure

### C/C++ Declaration

```
RECOIQA_API long IQ_GetMeasure(long SessionHandle, long Measure, long Index);
```

### C# Declaration

```
int IQA_GetMeasure(int Session,int Measure, int Index);
```

### Visual Basic Declaration

```
Function IQA_GetMeasure (ByVal Session As Long, ByVal Measure As Long, ByVal Index As Long) As
```

---

Long

### **Visual Basic .NET Declaration**

```
Function IQA_GetMeasure(ByVal Session As Integer,  
ByVal Measure As Integer, ByVal Index As Integer)  
As Integer
```

### **Delphi Declaration**

```
function  
IQA_GetMeasure(SessionHandle:Integer;Measure:Integer;  
Index:Integer):Integer; stdcall;
```

### **Java Declaration**

```
int IQA_GetMeasure(int SessionHandle, int Measure,  
int Index);
```

### **Description**

Get a measure computed by quality control checking.

### **Parameters**

*SessionHandle (in)* - the session handle to use. Its the value returned by library initialization function [IQA\\_Init](#).

*Measure (in)*: the index of measure to get. Allowed values are:

- 1:Undersize Image Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 2:Folded Corners Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 3:Folded Edges Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 4:Framing Error Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 5:Document Skew Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 6:Oversize Image Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 7:PiggyBack Document Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 8:Image Too Light Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.
- 9:Image Too Dark Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.

10:Horizontal Streak Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
11:Below Compressed Size Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
12:Above Compressed Size Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
13:Spot Noise Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
14:Front Rear Dimension Mismatch Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
15:Carbon Strip Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
16:Out Of Focus Flag - 0=Not Measured, 1=Defect Present, 2=Defect Absent.  
100:Width of image in 1/100 of inch.  
101:Height of image in 1/100 of inch.  
102:Left Edge Overscan length in 1/100 of inch.  
103:Right Edge Overscan length in 1/100 of inch.  
104:Top Edge Overscan length in 1/100 of inch.  
105:Bottom Edge Overscan length in 1/100 of inch.  
106:Top Left Folded Corner X coordinate if any  
107:Top Left Folded Corner Y coordinate if any  
108:Top Right Folded Corner X coordinate if any  
109:Top Right Folded Corner Y coordinate if any  
110:BottomLeftFoldedCorner X coordinate if any  
111:BottomLeftFoldedCorner Y coordinate if any  
112:BottomRightFoldedCorner X coordinate if any  
113:BottomRightFoldedCorner Y coordinate if any  
114:Top Edge Folded Zone Count  
115:Top Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
116:Top Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
117:Top Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
118:Top Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)  
119:Bottom Edge Folded Zone Count  
120:Bottom Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
121:Bottom Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
122:Bottom Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
123:Bottom Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)

- 124:Right Edge Folded Zone.Count
- 125:Right Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 126:Right Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 127:Right Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 128:Right Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 129:Left Edge Folded Zone Count
- 130:Left Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 131:Left Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 132:Left Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 133:Left Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)
- 134:SkewAngle in 1/10 of degree.
- 135:Percent Brighthness in 1/10 of percent.
- 136:Percent Contrast in 1/10 of percent.
- 137:Percent Black Pixels in 1/10 of percent.
- 138:Average Spot Noise Count
- 139:Image Focus Score

*Index*: integer value representing the index of measure to find, if the measure is an array of values representing top, right, bottom, left rects.

### Return values

The measured value or 0xFFFFFFFF if some error occurred.

## 4.7 IQA\_GetMeasureName

### C/C++ Declaration

```
RECOIQA_API long IQ_GetMeasureName(long
SessionHandle, long Measure, long Index, char*
Buffer);
```

### C# Declaration

```
int IQA_GetMeasureName(int Session, int Measure,
int Index, StringBuilder MeasureName);
```

## **Visual Basic Declaration**

```
Function IQA_GetMeasureName (ByVal Session As Long, ByVal Measure As Long, ByVal Index As Long, ByVal Buffer As String) As Long
```

## **Visual Basic .NET Declaration**

```
Function IQA_GetMeasureName(ByVal Session As Integer, ByVal Measure As Integer, ByVal Index As Integer, ByVal Buffer As StringBuilder) As Integer
```

## **Delphi Declaration**

```
function  
IQA_GetMeasureName(SessionHandle:Integer;Measure:Integer;Index:Integer;MeasureName:PAnsiChar):Integer;  
stdcall;
```

## **Java Declaration**

```
int IQA_GetMeasureName(int SessionHandle, int Measure, int Index, Memory Name);
```

## **Description**

Get a textual description of measure computed by quality control checking.

## **Parameters**

*SessionHandle (in)* - the session handle to use. Its the value returned by library initialization function [IQA\\_Init](#).

*Measure (in)*: the index of measure to get. Allowed values are:

- 1:Undersize Image Flag
- 2:Folded Corners Flag
- 3:Folded Edges Flag
- 4:Framing Error Flag
- 5:Document Skew Flag
- 6:Oversize Image Flag
- 7:PiggyBack Document Flag
- 8:Image Too Light Flag
- 9:Image Too Dark Flag
- 10:Horizontal Streak Flag
- 11:Below Compressed Size Flag
- 12:Above Compressed Size Flag

13:Spot Noise Flag  
14:Front Rear Dimension Mismatch Flag  
15:Carbon Strip Flag  
16:Out Of Focus Flag  
100:Width of image  
101:Height of image  
102:Left Edge Overscan  
103:Right Edge Overscan  
104:Top Edge Overscan  
105:Bottom Edge Overscan  
106:Top Left Folded Corner X  
107:Top Left Folded Corner Y  
108:Top Right Folded Corner X  
109:Top Right Folded Corner Y  
110:BottomLeftFoldedCorner X  
111:BottomLeftFoldedCorner Y  
112:BottomRightFoldedCorner X  
113:BottomRightFoldedCorner Y  
114:Top Edge Folded Zone Count  
115:Top Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
116:Top Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
117:Top Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
118:Top Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)  
119:Bottom Edge Folded Zone Count  
120:Bottom Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
121:Bottom Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
122:Bottom Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
123:Bottom Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)  
124:Right Edge Folded Zone.Count  
125:Right Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
126:Right Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
127:Right Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
128:Right Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)  
129:Left Edge Folded Zone Count

130:Left Edge Folded Zone Rect Left coordinate (requires usage of index parameter from 0 to Zone Count-1)  
131:Left Edge Folded Zone Rect Top coordinate (requires usage of index parameter from 0 to Zone Count-1)  
132:Left Edge Folded Zone Rect Right coordinate (requires usage of index parameter from 0 to Zone Count-1)  
133:Left Edge Folded Zone Rect Bottom coordinate (requires usage of index parameter from 0 to Zone Count-1)  
134:SkewAngle.  
135:Percent Brightness  
136:Percent Contrast  
137:Percent Black Pixels  
138:Average Spot Noise Count  
139:Image Focus Score

*Index (in)*: integer value representing the index of measure to find, if the measure is an array of values representing top, right, bottom, left rects.

*Buffer (out)*: pointer to a buffer where the textual description of measure is copied. You have to allocate 80 characters for safety or you can call the function with the buffer as NULL to get the right number of chars to allocate.

#### Return values

The length of measure description, in chars.

## 4.8 LoadIQALibrary

### C/C++ Declaration

```
void LoadIQALibrary(void)
```

### Description

Load the IQA DLL library: you have to use this function one time before to use other API functions from Visual C++. This function is not required using the API from Delphi or Visual Basic

## 4.9 FreeIQALibrary

### C/C++ Declaration

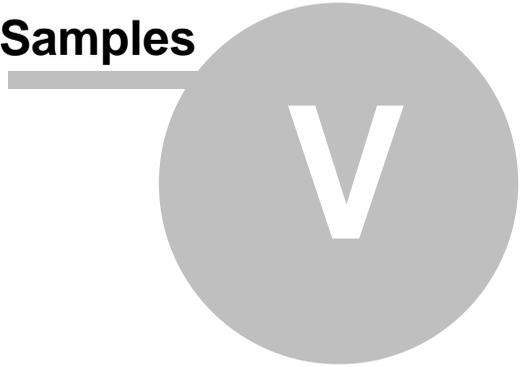
```
void FreeIQALibrary(void)
```

### Description

Unload the IQA DLL library: you have to use this function one time before to exit from your Visual C++ application. This function is not required using the API from Delphi or Visual Basic



**Samples**



## 5 Samples

### 5.1 Visual C++

This sample perform IQA on the image found in the clipboard, assumint it as front side of check and using default parametes.

```
#include "stdafx.h"
#include "recoiqa.c"
int main(int argc, char* argv[])
{
    int IQAsession, hBitmap, iReturnCode,
    MeasureValue, i;
    char buffer[1000];
    char MeasureName[255];
    // Load dynamically the IQA Library
    LoadIQALibrary();
    // Initialize the IQA engine
    IQAsession= IQA_Init("demo", "demo");
    // *****
    // IMPORTANT NOTICE
    // *****
    // YOU SHOULD LOAD HERE YOUR CHECK IMAGES,
    FRONT AND BACK,
    // OBTAINING THE DIB HANDLES TO PASS TO
    IQA_EXECUTE
    // YOU CAN GIVE A LOOK TO OUR IMAGING
    LIBRARY
    // THIS DEMO SIMPLY PASTE FROM CLIBOARD AN
    IMAGE THAT
    // YOU PREVIOUSLY HAVE TO COPY INTO
    CLIPBOARD FROM MSPAINT
    // OR OTHER IMAGING TOOL, JUST CONSIDERING
    IT AS THE FRONT
    // OF YOUR CHECK
    hBitmap = 0;
    // Check if a bitmpa is available in the
    clipboard
    if (IsClipboardFormatAvailable(CF_DIB))
    {
        // Open the clipboard
        ::OpenClipboard(NULL);
        // Get the DIB handle from clipboard
        hBitmap = (long)
        GetClipboardData(CF_DIB);
        // Perform processing on DIB handle
        (just on check fron side)
        iReturnCode =
```

---

```

IQA_Execute(IQAsession,hBitmap, 0, NULL);
    // Close the clipboard
    ::CloseClipboard();
    // Show flags (0=QC not executed
(unable to perform), 1=QC NOT passed (defect
present), 2=QC passed (defect absent)
    printf("*** FLAGS ***\n");
    i=1;
    while (i<=16) {
        MeasureName[0]=(char)0;

IQA_GetMeasureName(IQAsession,i,0,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i,0);
    sprintf(buffer,"%s =
%d\n",MeasureName, MeasureValue);
    printf(buffer);
    i++;
    }
    // Show measures
    printf("*** MEASURES ***\n");
    // Show all measures
    i=100;
    while (i<=139) {
        MeasureName[0]=(char)0;

IQA_GetMeasureName(IQAsession,i,0,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i,0);
    sprintf(buffer,"%s =
%d\n",MeasureName, MeasureValue);
    printf(buffer);
    // Check if the measure is the
counter of rectangular zones
        if ( (i==114) || (i==119)
|| (i==124) || (i==129) ) {
            int Counter =MeasureValue;
            // Show left, top, right
and bottom coordinate of each zone
            for (int j=0;
j<Counter;j++) {

IQA_GetMeasureName(IQAsession,i+1,j,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i+1,j);
        sprintf(buffer,"%s
=%d \n",MeasureName, MeasureValue);

```

---

```
IQA_GetMeasureName(IQAsession,i+2,j,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i+2,j);
    sprintf(buffer,"%s
=%d \n",MeasureName, MeasureValue);

IQA_GetMeasureName(IQAsession,i+3,j,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i+3,j);
    sprintf(buffer,"%s
=%d \n",MeasureName, MeasureValue);

IQA_GetMeasureName(IQAsession,i+4,j,MeasureName);
    MeasureValue
=IQA_GetMeasure(IQAsession,i+4,j);
    sprintf(buffer,"%s
=%d \n",MeasureName, MeasureValue);
        } // for
        // Skip already used zones
        i=i+4;
    } // if
    i++;
} // while
} // if
else
    MessageBox(NULL, "Unable to paste DIB from
clipboard", "ERROR", MB_OK);
    // Deinitialize the IQA session
    IQA_Done(IQAsession);
    // Free the IQA library because service
isnot required
    FreeIQALibrary();
    return 0;
} // main
```

## *Annotation*

