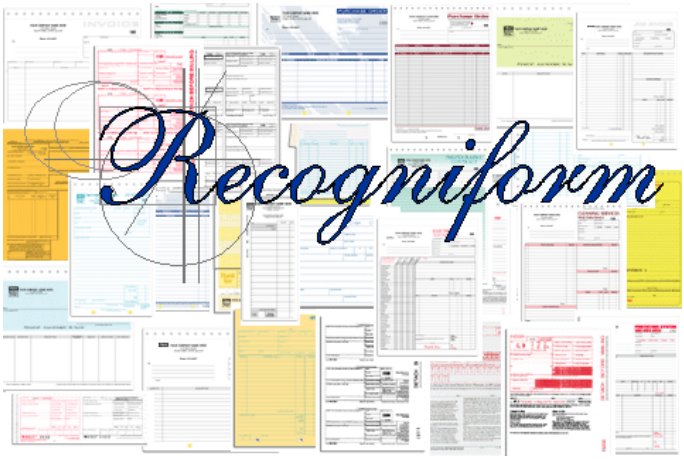


Software Development Kit

OCR

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

E-Mail : info@recogniform.com

Table of contents

Introduction 6

- Copyright ... 6
- License ... 6
- Overview ... 6

Usage 9

- Visual C++ ... 9
- C# ... 9
- Visual Basic ... 9
- Visual Basic .NET ... 9
- Delphi ... 9
- Java ... 9

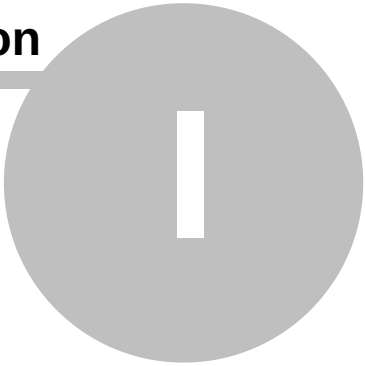
API References 11

- OCR_Init ... 11
- OCR_Done ... 12
- OCR_SetParameter ... 13
- OCR_RecognizeChar ... 14
- OCR_RecognizeWord ... 16
- OCR_RecognizeLine ... 17
- OCR_RecognizeParagraph ... 18
- OCR_RecognizePage ... 20
- OCR_GetResultLen ... 21
- OCR_GetResultConfidence ... 22
- OCR_GetResultData ... 23
- OCR_GetCharacterConfidence ... 24
- OCR_GetCharacterData ... 25
- OCR_GetCharacterRect ... 26
- OCR_GetCharacterAlternativesCount ... 28
- OCR_GetCharactersCount ... 29
- OCR_LoadImage ... 30
- OCR_FreeImage ... 31
- LoadOCRLibrary ... 32
- FreeOCRLibrary ... 32

Sample **34**

Code Sample **34**

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.

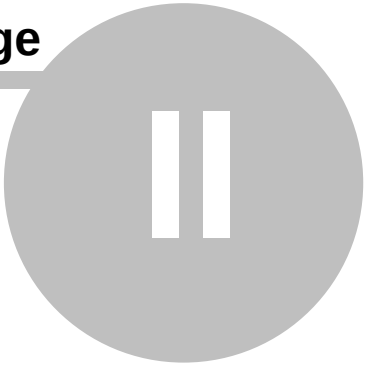
YOU CANNOT DISTRUBUTE SOFTWARE INCLUDING THIS SDK LIBRARY UNLESS YOU HAVE A WRITTEN AGREEMENT (ROYALTIES FREE OR ROYALTIES BASED) WITH RECOGNIFORM TECHNOLOGIES SPA

1.3 Overview

The library allows to recognize printed text from images acquired using a scanner. The character set supported includes all latin characters, with or without accents, digits, and common symbols. The system is omnifont, so it can virtually recognize all non decorative typeface:

The recognition process is fast and accurate and you can get too the confidence and the rectangle area of each recognized character.

Usage



2 Usage

2.1 Visual C++

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

2.2 C#

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

2.3 Visual Basic

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

2.4 Visual Basic .NET

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

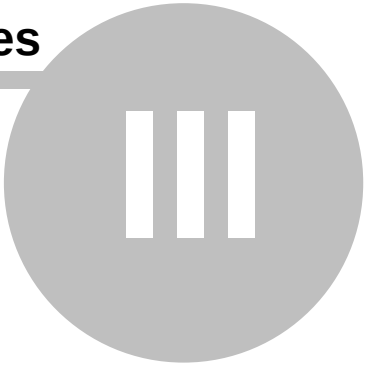
2.5 Delphi

You have to include the RECOOCRAPI.PAS in your program. Before to execute your application make sure the *RECOOCR.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 RECOOCRAPI.java in your program. Before to execute your application make sure the *RECOOCR.DLL* is available in your same .jar directory.

API References



3 API References

3.1 OCR_Init

C/C++ Declaration

```
__stdcall long OCR_Init(char* Name, char* Key);
```

C# Declaration

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

Visual Basic Declaration

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

Visual Basic .NET Declaration

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

Delphi Declaration

```
function OCR_Init(Company:PAnsiChar;
LicenseKey:PAnsiChar):Integer; stdcall;
```

Java Declaration

```
int OCR_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 wrang values the library is initialized in evaluation mode. The evaluation mode works exactly as normal mode but some time, when you call the recognition 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

Passwors (in) - then password string

Return values

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

3.2 OCR_Done

C/C++ Declaration

```
__stdcall void OCR_Done(long Session);
```

C# Declaration

```
void OCR_Done(int Session);
```

Visual Basic Declaration

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

Visual Basic .NET Declaration

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

Delphi Declaration

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

Java Declaration

```
void OCR_Done(int Session);
```

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

3.3 OCR_SetParameter

C/C++ Declaration

```
__stdcall long OCR_SetParameter(long Session, long
ParameterIndex, void* ParameterValue);
```

C# Declaration

```
int OCR_SetParameter(int Session, int
ParameterIndex, int ParameterValue);
```

Visual Basic Declaration

```
Function OCR_SetParameter(ByVal Session As
Integer, ByVal ParameterIndex As Integer, ByVal
ParameterValue As Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_SetParameter(ByVal Session As
Integer, ByVal ParameterIndex As Integer, ByVal
ParameterValue As Integer) As Integer
```

Delphi Declaration

```
function OCR_SetParameter(SessionHandle:Integer;
ParameterIndex:Integer;
ParameterValue:Pointer):Integer; stdcall;
```

Java Declaration

```
int OCR_SetParameter(int Session, int
ParameterIndex, int ParameterValue);
```

Description

This is the function to call to change default recognition parameters.

Parameters

SessionHandle (in) - the session handle to free
ParameterIndex (in) - the parameter number to change: can be 0 (allowed character set), 1 (numeric/alphanumeric mode), 2 (spot removal mode), 3 (max spot size), 4 (character split mode), 5 (similarity correction enabled/disabled), 6 (context correction

enabled/disabled), 7 (auto-zoning enable/disable)
ParameterValue (in) - the parameter value to set: allowed values changes according to *ParameterIndex.* If the parameter index is:
0 : you have to supply a pointer to an unicode string with all the recognizable characters. If the supplied pointer is null (0) then it will be used the full character set.
1 : you can specify simply a value 0, to enable alphanumeric recognition mode, and 1 to enable just numeric recognition mode.
2 : you can specify simply a value 0 to disable spot removal, 1 to enable smart spot removal end 2 to enable classical spot removal.
3 : you can specify the maximum diameter, in pixel, of spots to remove.
4 : you can specify simply a value 0 to disable character splitting, 1 to enable only “sure” character splitting and 2 to enable classical character splitting. In sure mode, the splitting is confirmed just if all split characters are recognized with success.
5 : you can specify 0 to disable similarity correction, 1 to enable similarity correction. Similarity correction allow to recognize unclassified characters finding shape similar characters already corrected classified.
6 : you can specify 0 to disable context correction, 1 to enable context correction. Context correction allows to auto discriminate between similar character, as “1” (one) and “l” (ell) , looking for adjacent characters.
7 : you can specify 0 to disable auto zoning, 1 to enable auto zoning. Auto-zoning is used when you have to read a full page, attempting to preserve columnar order.

Return values

1 if the parameter was set with success, 0 if some error occurred (bad parameter index or bad parameter value)

3.4 OCR_RecognizeChar

C/C++ Declaration

```
__stdcall long OCR_RecognizeChar(long Session,
long DIBIn, long Options);
```

C# Declaration

```
int OCR_RecognizeChar(int Session, int DIBIn, int
Options);
```

Visual Basic Declaration

```
Function OCR_RecognizeChar(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_RecognizeChar(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Delphi Declaration

```
function OCR_RecognizeChar(SessionHandle:Integer;
DIBHandle:Integer; Options:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_RecognizeChar(int Session,int DIBIn, int
Options);
```

Description

This is the function performing the recognition on a single char: you have to call this function after library initialization to perform the optical character recognition. The DIB handle haves to contain a single character.

Parameters

SessionHandle (in) - the session handle to use
DIBIn (in) - the handle of the DIB to recognize
Options (in) - the options to use (not used at the moment)

Return values

The number of characters recognized if success, 0 otherwise.

Notes

The library works with standard DIBs images. Refer to Microsoft documentation for additional info about Device Independent

Bitmaps. You can use color or grayscale images to get better results.

3.5 OCR_RecognizeWord

C/C++ Declaration

```
__stdcall long OCR_RecognizeWord(long Session,
long DIBIn, long Options);
```

C# Declaration

```
int OCR_RecognizeWord(int Session, int DIBIn, int
Options);
```

Visual Basic Declaration

```
Function OCR_RecognizeWord(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_RecognizeWord(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Delphi Declaration

```
function OCR_RecognizeWord(SessionHandle:Integer;
DIBHandle:Integer; Options:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_RecognizeWord(int Session,int DIBIn, int
Options);
```

Description

This is the function performing the recognition: you have to call this function after library initialization to perform the optical character recognition. The DIB handle haves to contain a single word.

Parameters

SessionHandle (in) - the session handle to use
DIBIn (in) - the handle of the DIB to recognize
Options (in) - the options to use (not used at the moment)

Return values

The number of characters recognized if success, 0 otherwise.

Notes

The library works with standard DIBs images. Refer to Microsoft documentation for additional info about Device Independent Bitmaps. You can use color or grayscale images to get better results.

3.6 OCR_RecognizeLine

C/C++ Declaration

```
__stdcall long OCR_RecognizeLine(long Session,
long DIBIn, long Options);
```

C# Declaration

```
int OCR_RecognizeLine(int Session, int DIBIn, int
Options);
```

Visual Basic Declaration

```
Function OCR_RecognizeLine(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_RecognizeLine(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Delphi Declaration

```
function OCR_RecognizeLine(SessionHandle:Integer;
DIBHandle:Integer; Options:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_RecognizeLine(int Session,int DIBIn, int
Options);
```

Description

This is the function performing the recognition: you have to call this function after library initialization to perform the optical character recognition. The DIB handle haves to contain a single line of text.

Parameters

SessionHandle (in) - the session handle to use
DIBIn (in) - the handle of the DIB to recognize
Options (in) - the options to use (not used at the moment)

Return values

The number of characters recognized if success, 0 otherwise.

Notes

The library works with standard DIBs images. Refer to Microsoft documentation for additional info about Device Independent Bitmaps. You can use color or grayscale images to get better results.

3.7 OCR_RecognizeParagraph

C/C++ Declaration

```
__stdcall long OCR_RecognizeParagraph(long
Session, long DIBIn, long Options);
```

C# Declaration

```
int OCR_RecognizeParagraph(int Session, int DIBIn,
int Options);
```

Visual Basic Declaration

```
Function OCR_RecognizeParagraph(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_RecognizeParagraph(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Delphi Declaration

```
function
OCR_RecognizeParagraph(SessionHandle:Integer;
DIBHandle:Integer; Options:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_RecognizeParagraph(int Session, int DIBIn,
int Options);
```

Description

This is the function performing the recognition: you have to call this function after library initialization to perform the optical character recognition. The DIB handle haves to contain a text paragraph.

Parameters

SessionHandle (in) - the session handle to use
DIBIn (in) - the handle of the DIB to recognize
Options (in) - the options to use (not used at the moment)

Return values

The number of characters recognized if success, 0 otherwise.

Notes

The library works with standard DIBs images. Refer to Microsoft documentation for additional info about Device Independent Bitmaps. You can use color or grayscale images to get better results.

3.8 OCR_RecognizePage

C/C++ Declaration

```
__stdcall long OCR_RecognizePage(long Session,
long DIBIn, long Options);
```

C# Declaration

```
int OCR_RecognizePage(int Session, int DIBIn, int
Options);
```

Visual Basic Declaration

```
Function OCR_RecognizePage(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_RecognizePage(ByVal Session As
Integer, ByVal DIBIn As Integer, ByVal Options As
Integer) As Integer
```

Delphi Declaration

```
function OCR_RecognizePage(SessionHandle:Integer;
DIBHandle:Integer; Options:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_RecognizePage(int Session,int DIBIn, int
Options);
```

Description

This is the function performing the recognition: you have to call this function after library initialization to perform the optical character recognition. The DIB handle haves to contain a full page.

Parameters

SessionHandle (in) - the session handle to use
DIBIn (in) - the handle of the DIB to recognize
Options (in) - the options to use (not used at the moment)

Return values

The number of characters recognized if success, 0 otherwise.

Notes

The library works with standard DIBs images. Refer to Microsoft documentation for additional info about Device Independent Bitmaps. You can use color or grayscale images to get better results.

3.9 OCR_GetResultLen

C/C++ Declaration

```
__stdcall long OCR_GetResultLen(long Session);
```

C# Declaration

```
int OCR_GetResultLen(int Session);
```

Visual Basic Declaration

```
Function OCR_GetResultLen(ByVal Session As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_GetResultLen(ByVal Session As
Integer) As Integer
```

Delphi Declaration

```
function
OCR_GetResultLen(SessionHandle:Integer):Integer;
stdcall;
```

Java Declaration

```
int OCR_GetResultLen(int Session);
```

Description

This is the function to call after the recognition process to get the number of characters recognized in the image, spaces included

Parameters

SessionHandle (in) - the session handle to use

Return values

The numbers of recognized characters, spaces included

3.10 OCR_GetResultConfidence

C/C++ Declaration

```
__stdcall long OCR_GetResultConfidence(long
Session);
```

C# Declaration

```
int OCR_GetResultConfidence(int Session);
```

Visual Basic Declaration

```
Function OCR_GetResultConfidence(ByVal Session As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_GetResultConfidence(ByVal Session As
Integer) As Integer
```

Delphi Declaration

```
function
OCR_GetResultConfidence(SessionHandle:Integer):Int
eger; stdcall;
```

Java Declaration

```
int OCR_GetResultConfidence(int Session);
```

Description

This is the function to call after the recognition process to get the confidence attributed to the last recognition, in the range 0..100.

Parameters

SessionHandle (in) - the session handle to use

Return values

The confidence level from 0 to 100

Notes

The confidence reported from this function is the confidence of the char with lowest confidence.

3.11 OCR_GetResultData

C/C++ Declaration

```
__stdcall long OCR_GetResultData(long Session,
char* Buffer);
```

C# Declaration

```
void OCR_GetResultData(int Session, StringBuilder
Buffer);
```

Visual Basic Declaration

```
Sub OCR_GetResultData(ByVal Session As Integer,
ByVal Buffer As String)
```

Visual Basic .NET Declaration

```
Sub OCR_GetResultData(ByVal Session As Integer,
ByVal Buffer As StringBuilder)
```

Delphi Declaration

```
function OCR_GetResultData(SessionHandle:Integer;
Buffer:PAnsiChar):Integer; stdcall;
```

Java Declaration

```
int OCR_GetResultData(int Session,Memory Buffer);
```

Description

This is the function to call after the recognition process to get the

chars recognized.

Parameters

SessionHandle (in) - the session handle to use
CharBuffer (out) - the buffer where will be moved the characters recognized. You have to allocate enough space.

Return values

n/a

3.12 OCR_GetCharacterConfidence

C/C++ Declaration

```
__stdcall void OCR_GetCharacterConfidence(long
Session, long CharIndex, long CharAlt, long*
Buffer);
```

C# Declaration

```
void OCR_GetCharacterConfidence(int Session, int
CharIndex, int CharAlt, ref int Buffer);
```

Visual Basic Declaration

```
Sub OCR_GetCharacterConfidence(ByVal Session As
Integer, ByVal CharIndex As Integer, ByVal CharAlt
As Integer, ByRef Buffer As Integer)
```

Visual Basic .NET Declaration

```
Sub OCR_GetCharacterConfidence(ByVal Session As
Integer, ByVal CharIndex As Integer, ByVal CharAlt
As Integer, ByRef Buffer As Integer)
```

Delphi Declaration

```
procedure
OCR_GetCharacterConfidence(SessionHandle:Integer;
CharIndex:Integer; CharAlternative:Integer; Var
Confidence:Integer); stdcall;
```

Java Declaration

```
void OCR_GetCharacterConfidence(int Session, int
```

```
CharIndex, int CharAlt, Memory Buffer);
```

Description

Retrieves the confidence of one of possible character alternative recognized. For each character you can get a variable number of alternatives.

Parameters

SessionHandle (in) - the session handle to use
CharIndex (in) - the index of the character . First character is 0, last is the OCR_GetCharactersCount-1.
CharAlternative (in) the index of the alternative, from 0 (hi confidence) to OCR_GetCharacterAlternativesCount-1 (low confidence).
Confidence (out) - the buffer where will be moved the confidence of characters. The value is in the range 0 (low) to 100 (hi).

Return values

n/a

3.13 OCR_GetCharacterData

C/C++ Declaration

```
__stdcall void OCR_GetCharacterData(long Session,
long CharIndex, long CharAlt, char* Buffer);
```

C# Declaration

```
void OCR_GetCharacterData(int Session, int
CharIndex, int CharAlt, string Buffer);
```

Visual Basic Declaration

```
Sub OCR_GetCharacterData(ByVal Session As Integer,
ByVal CharIndex As Integer, ByVal CharAlt As
Integer, ByVal Buffer As String)
```

Visual Basic .NET Declaration

```
Sub OCR_GetCharacterData(ByVal Session As Integer,
ByVal CharIndex As Integer, ByVal CharAlt As
Integer, ByVal Buffer As String)
```

Delphi Declaration

```
procedure
OCR_GetCharacterData(SessionHandle:Integer;
CharIndex:Integer; CharAlternative:Integer; Var
Value:Char); stdcall;
```

Java Declaration

```
void OCR_GetCharacterData(int Session,    int
CharIndex, int CharAlt, Memory Buffer);
```

Description

Retrieves the ascii value of one of possible character alternative recognized. For each character you can get a variable number of alternatives.

Parameters

SessionHandle (in) - the session handle to use
CharIndex (in) - the index of the character . First character is 0, last is the OCR_GetCharactersCount-1.
CharAlternative (in) the index of the alternative, from 0 (hi confidence) to OCR_GetCharacterAlternativesCount-1 (low confidence).
Value (out) - the buffer where will be moved the ASCII value of character.

Return values

n/a

3.14 OCR_GetCharacterRect

C/C++ Declaration

```
__stdcall void OCR_GetCharacterRect(long Session,
long CharIndex, long* Left, long* Top, long*
Right,long* Bottom);
```

C# Declaration

```
void OCR_GetCharacterRect(int Session, int
CharIndex, ref int Left, ref int Top, ref int
```

```
Right, ref int Bottom);
```

Visual Basic Declaration

```
Sub OCR_GetCharacterRect(ByVal Session As Integer,
ByVal CharIndex As Integer, ByRef Left As Integer,
ByRef Top As Integer, ByRef Right As Integer,
ByRef Bottom As Integer)
```

Visual Basic .NET Declaration

```
Sub OCR_GetCharacterRect(ByVal Session As Integer,
ByVal CharIndex As Integer, ByRef Left As Integer,
ByRef Top As Integer, ByRef Right As Integer,
ByRef Bottom As Integer)
```

Delphi Declaration

```
procedure
OCR_GetCharacterRect(SessionHandle:Integer;
CharIndex:Integer; Var
Left,Top,Right,Bottom:Integer); stdcall;
```

Java Declaration

```
void OCR_GetCharacterRect(int Session, int
CharIndex,  Memory Left,  Memory Top,  Memory
Right,  Memory Bottom);
```

Description

Retrieves the rectangle coordinates of one of possible character alternative recognized. For each character you can get a variable number of alternatives.

Parameters

SessionHandle (in) - the session handle to use
CharIndex (in) - the index of the character . First character is 0, last is the OCR_GetCharactersCount-1.
Left (out) - the buffer where will be moved the left position of the rect containing the character.
Top (out) - the buffer where will be moved the top position of the rect containing the character.
Right (out) - the buffer where will be moved the right position of the

rect containing the character.
Bottom (out) - the buffer where will be moved the bottom position of the rect containing the character.

Return values

n/a

3.15 OCR_GetCharacterAlternativesCount

C/C++ Declaration

```
__stdcall long
OCR_GetCharacterAlternativesCount(long Session,
long CharIndex);
```

C# Declaration

```
int OCR_GetCharacterAlternativesCount(int Session,
int CharIndex ) ;
```

Visual Basic Declaration

```
Function OCR_GetCharacterAlternativesCount(ByVal
Session As Integer, ByVal CharIndex As Integer) As
Integer
```

Visual Basic .NET Declaration

```
Function OCR_GetCharacterAlternativesCount(ByVal
Session As Integer, ByVal CharIndex As Integer) As
Integer
```

Delphi Declaration

```
function
OCR_GetCharacterAlternativesCount(SessionHandle:In
teger; CharIndex:Integer):Integer; stdcall;
```

Java Declaration

```
long OCR_GetCharacterAlternativesCount(int
Session, long CharIndex);
```

Description

Retrieves the number of alternatives available for each recognized character.

Parameters

SessionHandle (in) - the session handle to use
CharIndex (in) - the index of the character . First character is 0, last is the OCR_GetCharactersCount-1.

Return values

The numbers of alternative interpretations available, spaces excluded.

3.16 OCR_GetCharactersCount

C/C++ Declaration

```
__stdcall long OCR_GetCharactersCount(long
Session);
```

C# Declaration

```
int OCR_GetCharactersCount(int Session);
```

Visual Basic Declaration

```
Function OCR_GetCharactersCount(ByVal Session As
Integer) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_GetCharactersCount(ByVal Session As
Integer) As Integer
```

Delphi Declaration

```
function
OCR_GetCharactersCount(SessionHandle:Integer):Inte
ger; stdcall;
```

Java Declaration

```
long OCR_GetCharactersCount(int Session);
```

Description

This is the function to call after the recognition process to get the number of characters recognized in the image, spaces included.

Parameters

SessionHandle (in) - the session handle to use

Return values

The numbers of recognized characters, spaces included.

3.17 OCR_LoadImage

C/C++ Declaration

```
__stdcall long OCR_LoadImage(long Session, char*
FileName);
```

C# Declaration

```
int OCR_LoadImage(int Session, string FileName);
```

Visual Basic Declaration

```
Function OCR_LoadImage(ByVal Session As Integer,
ByVal FileName As String) As Integer
```

Visual Basic .NET Declaration

```
Function OCR_LoadImage(ByVal Session As Integer,
ByVal FileName As String) As Integer
```

Delphi Declaration

```
function OCR_LoadImage(SessionHandle:Integer;
FileName:PChar):Integer; stdcall;
```

Java Declaration

```
int OCR_LoadImage(int Session, String FileName);
```

Description

This function allows to load an image from file obtaining a DIB handle. Supported files formats are TIF, JPG, PNG and BMP

Parameters

SessionHandle (in) - the session handle to use
FileName (in) - the name of the file to load

Return values

The DIB handle with the image inside the file, 0 if an error occurred.

3.18 OCR_FreeImage

C/C++ Declaration

```
__stdcall long OCR_FreeImage(long Session, long
DIBHandle);
```

C# Declaration

```
void OCR_FreeImage(int Session, int DIBHandle);
```

Visual Basic Declaration

```
Sub OCR_FreeImage(ByVal Session As Integer, ByVal
DIBHandle As Integer)
```

Visual Basic .NET Declaration

```
Sub OCR_FreeImage(ByVal Session As Integer, ByVal
DIBHandle As Integer)
```

Delphi Declaration

```
procedure OCR_FreeImage(SessionHandle:Integer;
DIBHandle:integer); stdcall;
```

Java Declaration

```
int OCR_FreeImage(int Session, int DIBHandle);
```

Description

This function allows to remove from memory an image previously loaded from file.

Parameters

SessionHandle (in) - the session handle to use
DIBHandle (in) - the handle of the DIB to free

Return values

n/a

3.19 LoadOCRLibrary

C/C++ Declaration

```
long LoadOCRLibrary();
```

Description

Load the OCR DLL library: you have to use this function one time before to use other API functions

3.20 FreeOCRLibrary

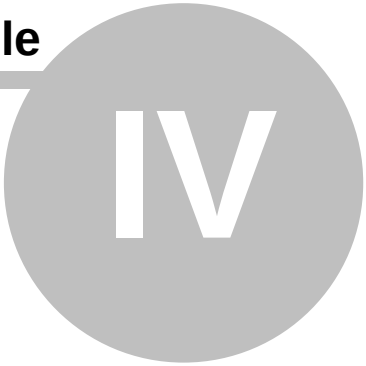
C/C++ Declaration

```
void FreeOCRLibrary();
```

Description

Unload the OCR DLL library: you have to use this function one time before to exit from your application.

Sample



4 Sample

4.1 Code Sample

```
#include "stdafx.h"
#include "recoocr.c"

int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE
hPrevInstance,LPSTR lpCmdLine, int nCmdShow)
{
 // Load dynamically the library
 LoadOCRLibrary();

 // Init the OCR session
 int Session= OCR_Init("demo", "demo");

 long hBitmap;

 char RecognizedString[255];

 // Check if a DIB is available in clipboard
 bool bAvail= IsClipboardFormatAvailable(CF_DIB);

 hBitmap = 0;

 if (bAvail)
 {
   // Open the Clipboard
   ::OpenClipboard(NULL);

   // Retrieve the DIB from clipboard
   hBitmap = (long) GetClipboardData(CF_DIB);

   // Recognize a line of text from the DIB
   int nChar= OCR_RecognizeLine(Session, (long) hBitmap, 0);

   // Close the Clipboard
   ::CloseClipboard();

   // Check if there are recognized chars
   if (nChar>0)
   {
    // Retrieve the recognized characters
    nChar = OCR_GetResultData(Session, RecognizedString);

    // Make the buffer a null terminated string
    RecognizedString[nChar]=0;

    // Show the recognized characters
    MessageBox(NULL, RecognizedString, "RESULT", MB_OK);
   }
   // Show an error message
   else MessageBox(NULL, "No data recognized !", "ERROR",
MB_OK);
  }
  // Show an error message
  else MessageBox(NULL, "Unable to paste DIB", "ERROR",
MB_OK);
```

```
 // Close the session
 OCR_Done(Session);

 // Unload the library
 FreeOCRLibrary();

 return 0;
}
```

Annotation