

# SDK Java API

Scanner Library Functions .....	4
1.1.1. Calibrate .....	4
1.1.2. CalibrateEx .....	5
1.1.3. CleanScanner .....	5
1.1.4. GetProgress .....	6
1.1.5. GetScannerType .....	6
1.1.6. InitScanLib .....	7
1.1.7. IsPaperOn .....	7
1.1.8. IsScannerValid .....	7
1.1.9. IsNeedCalibration .....	8
1.1.10. ResetImage .....	9
1.1.11. ScanToFile .....	9
1.1.12. ScanToFileEx .....	10
1.1.13. ScanToFileJ .....	10
1.1.14. ScanToFileBack .....	11
1.1.15. ScanToFileBackEx .....	12
1.1.16. ScanToFileBackJ .....	12
1.1.17. ScannerColorScheme .....	13
1.1.18. SetResolution .....	13
1.1.19. SetScanSize .....	14
1.1.20. UnInitSDK .....	15
1.1.21. SetDuplex .....	15
1.1.22. GetDuplex .....	15
1.1.23. DefaultScanner .....	16
1.1.24. UseFixedModel .....	17
1.1.25. GetDeviceSerialNumber .....	17
1.1.26. GetLibVersion .....	18
1.1.27. GetPressedButton .....	18
IdCard Library Functions .....	18
1.1.28. CountrySupportIDAutoDetect .....	18
1.1.29. DetectState .....	19
1.1.30. GetCntryNameById .....	20
1.1.31. GetCountryByState .....	20
1.1.32. GetIDFace .....	21
1.1.33. GetIDFaceEx .....	22
1.1.34. GetIDFaceDuplex .....	22
1.1.35. GetFirstCntry .....	23
1.1.36. GetFirstStateByCntry .....	24
1.1.37. GetNextCntry .....	24
1.1.38. GetNextStateByCntry .....	25
1.1.39. GetIDSignature .....	25
1.1.40. GetIDSignatureDuplex .....	26
1.1.41. GetStateNameById .....	27
1.1.42. GetIDStateShort .....	27
1.1.43. InitIdLib .....	28

1.1.44.	ProcessState.....	29
1.1.45.	GetCntryNameByld .....	29
1.1.46.	SetIDRegion.....	30
1.1.47.	SetIDRegionDetectionSequence.....	30
1.1.48.	GetFirstIDRegion .....	31
1.1.49.	GetNextIDRegion .....	31
1.1.50.	GetRegionNameByld .....	32
1.1.51.	GetRegionByCountry .....	32
1.1.52.	GetFirstCountryInRegion.....	33
1.1.53.	GetNextCountryInRegion .....	33
1.1.54.	GetIDFacelmgBuffer .....	34
1.1.55.	GetIDSignaturelmgBuffer .....	35
1.1.56.	DetectProcessAndCompare .....	36
1.1.57.	DetectProcessAndCompare2.....	36
1.1.58.	DetectProcessDuplex.....	37
1.1.59.	ResetIDFields .....	37
1.1.60.	GetIDImageAngle.....	38
1.1.61.	SetIDDatesFormat .....	38
1.1.62.	Data Functions.....	38
Barcode Library Functions .....		40
1.1.63.	InitBCLib .....	40
1.1.64.	ProcBarcode .....	41
1.1.65.	RefreshBC .....	41
1.1.66.	BC4DigitsYearFormat .....	42
1.1.67.	GetBCRawText .....	42
1.1.68.	Data Functions.....	43
Passport Library Functions .....		43
1.1.69.	InitPassportLib .....	43
1.1.70.	ProcessPassport.....	44
1.1.71.	ResetPassportData .....	44
1.1.72.	GetPassportMRZChecksumVerified.....	45
1.1.73.	GetPassportFaceEx .....	45
1.1.74.	GetPassportFacelmgBuffer .....	46
1.1.75.	GetPassportSignature .....	46
1.1.76.	GetPassportSignaturelmgBuffer.....	47
1.1.77.	Data Functions.....	47
ImageCtrl Library Functions.....		48
1.1.78.	InitImageLib .....	48
1.1.79.	GetImageColor.....	49
1.1.80.	Rotatelmage .....	49
1.1.81.	ReformatImage .....	51
1.1.82.	GetImageBufferData .....	52
1.1.83.	GetImageBufferBackData .....	53
1.1.84.	SetMainImage.....	53
Appendix A: Definitions.....		54
Appendix B – Supported States for Detection .....		67

## Scanner Library Functions

Scanner library is used to scan documents and save their image into external bitmap file and to the internal image buffer. The library also saves the scanned image to an external bitmap file. The library sets and retrieves the scanners properties (such as scanning size, resolution and color scheme).

### Calibrate

#### Format

**short Calibrate ()**

#### Return value

**LICENSE\_INVALID** – Library was not initialized with proper license.  
**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.  
**SLIB\_ERR\_INVALID\_SCANNER** – The attached scanner is invalid.  
**SLIB\_FALSE** – The operation failed (No calibration card was found)  
**SLIB\_TRUE** – Operation succeeded.

#### Remarks

This function calibrates the scanner using the calibration card. The calibration creates calibration files and places them under Windows\twain\_32\CSSNxxx directory as follows:

**Scanner type:** ScanShell600

**Calibration file directory:** C:\Documents and Settings\UserName\Local Settings\Temp

**Calibration files:** CSSCANA6.clb

**Scanner type:** ScanShell800

**Calibration file directory:** C:\WINDOWS\twain\_32\CSSN800

**Calibration files:** PixOffG6.dat, PixOff6.dat, PixGanG6.dat, PixGan6.dat

**Scanner type:** ScanShell800

**Calibration file directory:** C:\WINDOWS\twain\_32\CSSN800N

**Calibration files:** PixOffG6.dat, PixOff6.dat, PixGanG6.dat, PixGan6.dat

**Scanner type:** ScanShell1000

**Calibration file directory:** C:\WINDOWS\twain\_32\SS1000

**Calibration files:** PxGanGF6.dat, PixGanF6.dat, PxOffGF6.dat, PixOffF6.dat, PixGanG6.dat, PixOffG6.dat, PixGan6.dat, PixOff6.dat

**Notes:**

ScanShell800, ScanShell800N and ScanShell1000 scanners write the calibration files on a directory that might has security restriction in NTFS drives. This may prevent users with insufficient security level from calibrating the scanner, which results with calibration failure. To remove the security from this location run the program **ResetSec.exe** **once** while you are logged with Administrator privileges.

This application can be obtained from:

<http://www.id-scan.com/FTP/Applications/tools/ResetSec.exe>

**CalibrateEx**

**Format**

**short CalibrateEx ()**

**Return value**

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_INVALID\_SCANNER** – The attached scanner is invalid.

**SLIB\_FALSE** – The operation failed (No calibration card was found)

**SLIB\_TRUE** – Operation succeeded.

**Remarks**

This function is similar to **Calibrate** function but it will display calibration progress bar dialog.

**CleanScanner**

**Format**

**short CleanScanner ()**

**Return value**

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_INVALID\_SCANNER** – The attached scanner is invalid (neither ScanShell800 or ScanShell800N).

**SLIB\_FALSE** – The operation failed.

**SLIB\_TRUE** – Operation succeeded.

**Remarks**

This function clean the scanner lens by running the cleaning pad (supplied in the scanner kit) back and fourth

**Notes:**

This function applies only to ScanShell800 and ScanShell800N models.

**GetProgress**

**Format**

```
short GetProgress ()
```

**Return value**

A value in the range 0-100 that describe the current scan progress.

**Remarks**

Call this function from a separate thread to obtain the scan progress.

**GetScannerType**

**Format**

```
short GetScannerType ()
```

**Return value**

The attached scanner's type:

- CSSN\_NONE**– No attached scanner was found.
- CSSN\_600**– ScanShell600 is attached to the PC.
- CSSN\_800**– ScanShell800 is attached to the PC.
- CSSN\_800N**– ScanShell800N is attached to the PC.
- CSSN\_1000**– ScanShell1000 is attached to the PC.
- CSSN\_2000**– ScanShell2000 is attached to the PC.
- CSSN\_2000N**– ScanShell2000N is attached to the PC.
- CSSN\_800E**– ScanShell800E is attached to the PC.
- CSSN\_800EN**– ScanShell800EN is attached to the PC.
- CSSN\_3000**– ScanShell3000 is attached to the PC.
- CSSN\_4000**– ScanShell4000 is attached to the PC.
- CSSN\_800G**– ScanShell800G is attached to the PC.
- CSSN\_5000**– ScanShell5000 is attached to the PC.
- CSSN\_IDR**– SnapShell is attached to the PC.
- CSSN\_800DX**– ScanShell800DX is attached to the PC.
- CSSN\_800DXN**– ScanShell800DXN is attached to the PC.
- CSSN\_FDA**– SnapShell is attached to the PC.
- CSSN\_WMD**– SnapShell is attached to the PC.
- CSSN\_TWN**– SnapShell is attached to the PC.

**Remarks**

Call this function to retrieve the attached scanner type.

## InitScanLib

### Format

```
short InitScanLib (String License)
```

### Parameters

License – Null terminated string that holds license key value.

### Return value

**LICENSE\_VALID:** License is valid and the library is ready to be used.

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**LICENSE\_EXPIRED:** License has expired. All scanner operations are disabled.

**SLIB\_ERR\_DRIVER\_NOT\_FOUND:** The scanner driver was not found. To fix this error re-install the scanner's driver. All scanner operations are disabled.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND:** The scanner is not connected to the PC. To fix this error make sure the scanner is connected and re-starts the. All scanner operations are disabled.

### Remarks

Use this function to initialize the scanner library. This function loads the scanner driver and initializes the internal image structure. This function must be called before calling any other function in the library.

## IsPaperOn

### Format

```
short IsPaperOn ()
```

### Return value

**SLIB\_FALSE** – No document was detected in the scanner's tray.

**SLIB\_TRUE** – Document is in tray.

### Remarks

This function accesses the document sensor on the scanner and return if a document was detected.

## IsScannerValid

## Format

```
short IsScannerValid ()
```

## Return value

**SLIB\_FALSE** – Scanner is attached to the PC and resend correctly.

**SLIB\_TRUE** – No Scanner is attached to the PC.

## Remarks

This function accesses the scanner to verify its connectivity to the PC.

## IsNeedCalibration

### Format

```
short NeedCalibration ()
```

## Return value

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_FALSE** – Scanner is calibrated.

**SLIB\_TRUE** – Scanner needs to be calibrated.

## Remarks

Retrieve if the scanner needs to be calibrated. This should be tested before every scan. Non-calibrated scanner may generate images with incorrect colors.

## See Also

[Calibrate\(\)](#),



## ResetImage

### Format

```
short ResetImage ()
```

### Remarks

Delete the internal image buffer.

## ScanToFile

### Format

```
short ScanToFile (String filePath)
```

### Parameters

filePath – Null terminated string that holds destination file name.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL**

**SLIB\_ERR\_SCANNER\_NOT\_FOUND**

**SLIB\_ERR\_HARDWARE\_ERROR**

**SLIB\_ERR\_PAPER\_FED\_ERROR**

**SLIB\_ERR\_SCANABORT**

**SLIB\_ERR\_NO\_PAPER**

**SLIB\_ERR\_PAPER\_JAM**

**SLIB\_ERR\_FILE\_IO\_ERROR**

**SLIB\_ERR\_PRINTER\_PORT\_USED**

**SLIB\_ERR\_OUT\_OF\_MEMORY**

### Remarks

Scan document to the internal image buffer and, in the same time, export it to local disk. Notice that the image should be scanned in True color and 300 dpi format for proper OCR recognition.

## ScanToFileEx

### Format

short ScanToFileEx (String [filePath](#))

### Parameters

filePath – Null terminated string that holds destination file name.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL** – Scanner fail to start.

**SLIB\_ERR\_HARDWARE\_ERROR** – Scanner hardware error.

**SLIB\_ERR\_PAPER\_FED\_ERROR** – Document was not fed enough in tray to create image file.

**SLIB\_ERR\_SCANABORT** – Scanning was aborted.

**SLIB\_ERR\_NO\_PAPER** – No paper in tray.

**SLIB\_ERR\_PAPER\_JAM** - Paper was jammed while scanner is running.

**SLIB\_ERR\_FILE\_IO\_ERROR** – Fail to create image file on disk.

**SLIB\_ERR\_PRINTER\_PORT\_USED** – N/A

**SLIB\_ERR\_OUT\_OF\_MEMORY** - Not enough memory to create temporary image.

### Remarks

This function works identically to ScanToFile, and in addition, the function display a modal progress bar while the scan is in progress.

## ScanToFileJ

### Format

short ScanToFileJ (String [filePath](#))

### Parameters

filePath – Null terminated string that holds destination file name.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL** – Scanner fail to start.

- SLIB\_ERR\_HARDWARE\_ERROR** – Scanner hardware error.
- SLIB\_ERR\_PAPER\_FED\_ERROR** – Document was not fed enough in tray to create image file.
- SLIB\_ERR\_SCANABORT** – Scanning was aborted.
- SLIB\_ERR\_NO\_PAPER** – No paper in tray.
- SLIB\_ERR\_PAPER\_JAM** - Paper was jammed while scanner is running.
- SLIB\_ERR\_FILE\_IO\_ERROR** – Fail to create image file on disk.
- SLIB\_ERR\_PRINTER\_PORT\_USED** – N/A
- SLIB\_ERR\_OUT\_OF\_MEMORY** - Not enough memory to create temporary image.

### Remarks

This function works identically to ScanToFile, and in addition, the function display a modal progress bar while the scan is in progress. This function is typically called when calling Jar file in web pages to see the progress bar.

### ScanToFileBack

#### Format

short ScanToFileBack (String [filePath](#))

#### Parameters

filePath – Null terminated string that holds destination file name.

#### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

- LICENSE\_INVALID** – Library was not initialized with proper license.
- SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.
- SLIB\_ERR\_SCANNER\_GENERAL\_FAIL**
- SLIB\_ERR\_SCANNER\_NOT\_FOUND**
- SLIB\_ERR\_HARDWARE\_ERROR**
- SLIB\_ERR\_PAPER\_FED\_ERROR**
- SLIB\_ERR\_SCANABORT**
- SLIB\_ERR\_NO\_PAPER**
- SLIB\_ERR\_PAPER\_JAM**
- SLIB\_ERR\_FILE\_IO\_ERROR**
- SLIB\_ERR\_PRINTER\_PORT\_USED**
- SLIB\_ERR\_OUT\_OF\_MEMORY**

### Remarks

Scan document to the internal image buffer and, in the same time, export it to local disk. Notice that the image should be scanned in True color and 300 dpi format for proper OCR recognition.

## ScanToFileBackEx

### Format

short ScanToFileBackEx (String [filePath](#))

### Parameters

filePath – Null terminated string that holds destination file name.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL** – Scanner fail to start.

**SLIB\_ERR\_HARDWARE\_ERROR** – Scanner hardware error.

**SLIB\_ERR\_PAPER\_FED\_ERROR** – Document was not fed enough in tray to create image file.

**SLIB\_ERR\_SCANABORT** – Scanning was aborted.

**SLIB\_ERR\_NO\_PAPER** – No paper in tray.

**SLIB\_ERR\_PAPER\_JAM** - Paper was jammed while scanner is running.

**SLIB\_ERR\_FILE\_IO\_ERROR** – Fail to create image file on disk.

**SLIB\_ERR\_PRINTER\_PORT\_USED** – N/A

**SLIB\_ERR\_OUT\_OF\_MEMORY** - Not enough memory to create temporary image.

### Remarks

This function works identically to ScanToFile, and in addition, the function display a modal progress bar while the scan is in progress.

## ScanToFileBackJ

### Format

short ScanToFileJ (String [filePath](#))

### Parameters

filePath – Null terminated string that holds destination file name.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL** – Scanner fail to start.

- SLIB\_ERR\_HARDWARE\_ERROR** – Scanner hardware error.
- SLIB\_ERR\_PAPER\_FED\_ERROR** – Document was not fed enough in tray to create image file.
- SLIB\_ERR\_SCANABORT** – Scanning was aborted.
- SLIB\_ERR\_NO\_PAPER** – No paper in tray.
- SLIB\_ERR\_PAPER\_JAM** - Paper was jammed while scanner is running.
- SLIB\_ERR\_FILE\_IO\_ERROR** – Fail to create image file on disk.
- SLIB\_ERR\_PRINTER\_PORT\_USED** – N/A
- SLIB\_ERR\_OUT\_OF\_MEMORY** - Not enough memory to create temporary image.

### Remarks

This function works identically to ScanToFile, and in addition, the function display a modal progress bar while the scan is in progress. This function is typically called when calling Jar file in web pages to see the progress bar.

## ScannerColorScheme

### Format

```
short SetScannerColor (short color)
```

### Parameters

- color – The color scheme of the scanned image. Available values are:
- BW** – Black and White (1 bit) image.
- GRAY** – 256 shades of gray image.
- TRUECOLORL** –24 bit image (default).

### Return value

- LICENSE\_INVALID** – Library was not initialized with proper license.
- SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.
- SLIB\_ERR\_BAD\_PARAM** – Bad parameter value.
- SLIB\_ERR\_NONE**– function successful.

### Remarks

Determine the color scheme of the scan.

## SetResolution

### Format

```
short SetScannerResolution (short resolution)
```

### Parameters

- resolution – The value of the needed image resolution.

### Return value

If function succeeds, the return value is **SLIB\_ERR\_NONE**

If function fails, the return number is may be one of the following:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_SCANNER\_GENERAL\_FAIL** – Scanner fail to start.

**SLIB\_ERR\_CANCELED\_BY\_USER**

**SLIB\_ERR\_HARDWARE\_ERROR** – Scanner hardware error.

**SLIB\_ERR\_PAPER\_FED\_ERROR** – Document was not fed enough in tray to create image file.

**SLIB\_ERR\_SCANABORT** – Scanning was aborted.

**SLIB\_ERR\_NO\_PAPER** – No paper in tray.

**SLIB\_ERR\_PAPER\_JAM** - Paper was jammed while scanner is running.

**SLIB\_ERR\_FILE\_IO\_ERROR** – Fail to create image file on disk.

**SLIB\_ERR\_PRINTER\_PORT\_USED** – N/A

**SLIB\_ERR\_OUT\_OF\_MEMORY** - Not enough memory to create temporary image.

### Remarks

Set the scanner resolution settings. Resolution value can be any integer in the range 50-600 (for 50-600 dpi). The resolution value is rounded to the upper value of the nearest 100's boundary. I.e., input value of 180 will be rounded by the function to 200, input bvalue of 201 will be rounded to 300, etc. Trying to set value outside this range will be rejected and the previous value will be used. The scanner default resolution is 300 dpi.

## SetScanSize

### Format

```
short SetScanSize (short width, short height)
```

### Parameters

width – The document width in milli-inch. The scanner default width is 220 (2.2")

height – The document height in milli-inch. . The scanner default height is 360 (3.6")

### Return value

**LICENSE\_INVALID** – Library was not initialized with proper license.

**SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.

**SLIB\_ERR\_NONE** - function successful.

### Remarks

Set the document scan boundaries size. Document image beyond these boundaries will not reflect in the image file. If set to -1 for both values scanner will work in auto scan siz (this supported with regular scanners only).

## UnInitSDK

### Format

```
short UnInitSDK()
```

### Parameters

None.

### Return

**SLIB\_UNLOAD\_FAILED\_BAD\_PARENT:** Cannot unload the driver since another application is using has load it.

**SLIB\_NOT\_INITIALIZED:** The driver is not found in the memory (hence can't n be unloaded). **SLIB\_ERR\_NONE**– Library unloaded successfully.

### Remarks

The scanner library (and all other SDK libraries) can serve a single application at a time. If you wish to run another application (that use the SDK) you must first unload the SDK from the memory using this function. If you fail to do so, you might run into a situation that both applications will attempt to access the scanner (and other SDK resources) – an operation that might hang the application.

Use this function to unload the scanner driver from the memory (hence, to reverse the operation of the function *InitScanLib*) before initializing the SDK from the other application.

## SetDuplex

### Format

```
short SetDuplex(short type)
```

### Parameters

type – 1 or 0 (true/false).

### Remarks

Setting this value activate the double side scan when using scanner model ScannShell3000D\ ScannShell800DX\ ScannShell800DXN.

## GetDuplex

## Format

**Short GetDuplex()**

## Parameters

None.

## Remarks

Returns the Duplex value. 1 or 0 (true/false).

## DefaultScanner

### Format

**void DefaultScanner (int type)**

## Parameters

type – Scanner type.

Can be one of the following values:

- 0: No Scanner.
- 1: ScanShell600.
- 2: ScanShell800.
- 3: ScanShell800N.
- 4: ScanShell1000.
- 5: ScanShell2000.
- 6: ScanShell2000N.
- 7: ScanShell800E.
- 8: ScanShell800EN.
- 9: ScanShell3000D.
- 10: ScanShell4000.
- 11: ScanShell800G.
- 12: ScanShell5000.
- 13: SnapShell (IDR)
- 14: ScannShell800DX
- 15: ScannShell800DXN
- 16: SnapShell (FDA)
- 17: SnapShell (WMD)
- 18: SnapShell (TWN)
- 19: CSSN\_PASS
- 20: CSSN\_RTE8K
- 21: CSSN\_TWAIN\_N
- 22: CSSN\_MAGTEK\_STX
- 23: CSSN\_CLBS



### Remarks

Setting this value will set the SDK to load faster because the Slib will try to init the scanner type that has been set in this value first.

### UseFixedModel

#### Format

```
void UseFixedModel (int type)
```

#### Parameters

type – Scanner type.

Can be one of the following values:

- 0: No Scanner.
- 1: ScanShell600.
- 2: ScanShell800.
- 3: ScanShell800N.
- 4: ScanShell1000.
- 5: ScanShell2000.
- 6: ScanShell2000N.
- 7: ScanShell800E.
- 8: ScanShell800EN.
- 9: ScanShell3000D.
- 10: ScanShell4000.
- 11: ScanShell800G.
- 12: ScanShell5000.
- 13: SnapShell (IDR)
- 14: ScannShell800DX
- 15: ScannShell800DXN
- 16: SnapShell (FDA)
- 17: SnapShell (WMD)
- 18: SnapShell (TWN)
- 19: CSSN\_PASS
- 20: CSSN\_RTE8K
- 21: CSSN\_TWAIN\_N
- 22: CSSN\_MAGTEK\_STX
- 23: CSSN\_CLBS

### Remarks

Setting this value will set the SDK work only with the given scanner type.

### GetDeviceSerialNumber

### Format

```
int GetDeviceSerialNumber ()
```

### Parameters

None

### Remarks

Gets the hardware serial number.

## GetLibVersion

### Format

```
String GetLibVersion()
```

### Parameters

None

### Remarks

Returns the Slib version.

## GetPressedButton

### Format

```
short GetPressedButton()
```

### Parameters

None.

### Remarks

Returns the button number that was pressed on the scanner.

## IdCard Library Functions

IdCard library is used to analyze and extract data from the recently scanned driver's license image that is stored in the internal image buffer. The library also supports the extraction of the face and signature images.

## CountrySupportIDAutoDetect

## Format

**short CountrySupportIDAutoDetect (short countryId)**

## Parameters

countryId – Constant value of the country

## Return value

**ID\_TRUE:** The country support state auto detection.

**ID\_FALSE:** The country does not support state auto detection.

## Remarks

The state auto-detection feature is not implemented on all the supported countries. Use this function to determine which countries can use the DetectState function.

## DetectState

### Format

**short DetectState(String emptyString)**

## Parameters

emptyString – Empty string ("").

## Return value

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**ID\_ERR\_USA\_TEMPLATES\_NOT\_FOUND:** The template database file for the USA states (*UsaIds.bin*) is missing. The file should be located in the SDK directory.

**INVALID\_INTERNAL\_IMAGE:** No internal image is loaded. This value returns when attempting to use this function without scanning an image first.

**ID\_ERR\_STATE\_NOT\_SUPPORTED:** The license image does not match any state template.

**ID\_ERR\_STATE\_NOT\_RECOGNIZED:** The license image does not match any state template.

If non of the above error values is returned, the function return the state id value.

## Remarks

Use this function to automatically detect the state type according to the image. If the function returns with non of the above error values then the return value is

the state id. This value can be assigned to the input parameter ***IdState*** in the function ***ProcessState*** for data extraction.

#### Remarks

This function analyze the image in the internal buffer and determine were the issuing state.

### GetCntryNameById

#### Format

**String GetCntryNameById(short countryID)**

#### Parameters

[in] countryID – Constant value of the country.

#### Return value

**ID\_ERR\_NO\_MATCH:** The input country id value does not match any country value.

**ID\_TRUE:** Found country constant that match and the input parameter **countryID**. The country string is copied to **szCountryName..**

**Country id number:** Matching country id value.

#### Remarks

This function converts between the country id numeric value (constant short) and its textual name (string). The conversion can be from constant to string or from string to constant. The function examines the value of countryID. If this parameter is equal to -1 then the function takes the state name from the parameter szCountryName and return the country id. If the string doe not match any of the country names the function returns **ID\_ERR\_NO\_MATCH**. If the parameter countryID is not -1 then the function load the string szCountryName with the state textual name and the function returns **ID\_TRUE**. If the countryID does not match any of the country constants then the function returns **ID\_ERR\_NO\_MATCH**.

The string compare are all case sensitive.

### GetCountryByState

#### Format

**short GetCountryByState(int stateld)**

### Parameters

stateId – Constant value that represent the state.

### Return value

**ID\_ERR\_NO\_MATCH:** The input state id value does not match any known state value.

**Country id number:** The country id value that contain the state.

### Remarks

Use this function to find to what country the state belong. If the input value does not match any known state then the function returns **ID\_ERR\_NO\_MATCH**.

## GetIDFace

### Format

```
short GetIDFace(String sourceFile, String destFile, short stateID)
```

### Parameters

SourceFile – Full name of the source driver’s license image file.

DestFile – Full name of the destination face image.

stateID – The state id.

### Return value

**ID\_ERR\_FILE\_OPEN:** Cannot open input image file.

**INVALID\_INTERNAL\_IMAGE:** Invalid internal image file.

**ID\_FALSE:** Image processing failed.

**ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE:** Destination file already exists and cannot be overwritten.

**ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION:** Copying face image file to destination file failed.

**ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND:** Extraction of the face image failed – could not locate the face rectangle.

**ID\_TRUE:** Function completed successfully.

### Remarks

Use this function to extract the face image from the document image. The source document image is last scanned document (stored in the internal image buffer). If the parameter **szSourceFile** is an empty string then the image is taken from the internal image buffer.

## GetIDFaceEx

### Format

```
GetIDFaceEx(String sourceFile, String destFile, short stateID, short ImageType)
```

### Parameters

szSourceFile – Full name of the source driver’s license image file.  
 szDestFile – Full name of the destination face image.  
 stateID – The state id.  
 ImageType – The face image type to extract .

### Return value

**ID\_ERR\_FILE\_OPEN:** Cannot open input image file.  
**INVALID\_INTERNAL\_IMAGE:** Invalid internal image file.  
**ID\_FALSE:** Image processing failed.  
**ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE:** Destination file already exists and cannot be overwritten.  
**ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION:** Copying face image file to destination file failed.  
**ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND:** Extraction of the face image failed – could not locate the face rectangle.  
**ID\_TRUE:** Function completed successfully.

### Remarks

This function is similar to GetFace function but can extract more then one face image from the card. The imageType value can be 0 to get the regular face image or 1 to get face image with background. This function with ImageType other then 0 value will work only for supported cards, for now only “Spain police” cards supported by this function.

## GetIDFaceDuplex

### Format

```
short GetIDFaceDuplex(String sourceFile, String backSourceFile, String destFile, short stateID)
```

### Parameters

szSourceFile – Full name of the source driver’s license image file.  
 szBackSourceFile – Full name of the source driver’s license image file second side.  
 szDestFile – Full name of the destination face image.  
 stateID – The state id.

### Return value

- ID\_ERR\_FILE\_OPEN:** Cannot open input image file.
- INVALID\_INTERNAL\_IMAGE:** Invalid internal image file.
- ID\_FALSE:** Image processing failed.
- ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE:** Destination file already exists and cannot be overwritten.
- ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION:** Copying face image file to destination file failed.
- ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND:** Extraction of the face image failed – could not locate the face rectangle.
- ID\_TRUE:** Function completed successfully.

### Remarks

This function is for Duplex scan support that gives you the option to get the face image without knowing where it is (front or back side). Duplex scanning is available only with Scanshell 800DX\800DXN\3000D,5000. Use this function to extract the face image from the document image. The source document is last scanned document (stored in the internal image buffer). If the parameter **szSourceFile** is an empty string then the image is taken from the internal image buffer.

## GetFirstCntry

### Format

```
short GetFirstCntry()
```

### Return value

The value of the first country in the countries list.

### Remarks

Use this function to get the first country in the countries list. Combining this function with the function *GetNextCntry* allow you to obtain the constant values of all supported countries. The countries constant values in the region ARE NOT always consecutive and should be obtained using the function *GetNextCntry*.

### See Also

[GetNextCntry\(\)](#)

## GetFirstStateByCntry

### Format

```
short GetFirstStateByCntry(short country)
```

### Parameters

country – Constant value of the country.

### Return value

**ID\_ERR\_NO\_MATCH:** Bad country constant.

**ID\_TRUE:** Function completed successfully.

First state in the country

### Remarks

Use this function to retrieve the first state constant in the country. The states constant values in the country ARE NOT always consecutive  
And should be obtained using the function *GetNextStateByCntry*.

### See Also

*GetNextStateByCntry()*

## GetNextCntry

### Format

```
short GetNextCntry ()
```

### Return value

- **ID\_ERR\_COUNTRY\_NOT\_INIT:** Returned if *GetFirstCntry* function was not called before.
- **ID\_ERR\_NO\_MATCH:** Returned if list has internal error.
- **ID\_ERR\_NO\_NEXT\_COUNTRY** – Returned if this country is the last country in the list.
- Return the next country constant.

### Remarks

Use this function to obtain the next country in the country list. To obtain the countries list call *GetFirstCntry* once to obtain the first country. Then countiuously call *GetNextCntry* until the value **ID\_ERR\_NO\_NEXT\_COUNTRY** is returned.



**See Also**

[GetFirstCntry\(\)](#)

**GetNextStateByCntry**

**Format**

```
short GetNextStateByCntry(short country)
```

**Parameters**

country – Constant value of the country.

**Return value**

- **ID\_ERR\_NO\_MATCH**: Bad country constant.
- **ID\_ERR\_COUNTRY\_NOT\_INIT**: Returned if *GetFirstCntry* function was not called before..
- **ID\_ERR\_NO\_NEXT\_STATE** – Returned if this state is the last state of the country state list .
- Return the next state constant.

**Remarks**

Use this function to obtain the next state in the list.

**See Also**

[GetFirstStateByCntry\(\)](#).

**GetIDSignature**

**Format**

```
short GetIDSignature(String sourceFile, String destFile, short stateID)
```

**Parameters**

SourceFile – Full name of the source driver’s license image file.  
 DestFile – Full name of the destination signature image.  
 stateID – The state id.

**Return value**

**ID\_ERR\_FILE\_OPEN**: Cannot open input image file.

**INVALID\_INTERNAL\_IMAGE:** Invalid internal image file.

**ID\_FALSE:** Image processing failed.

**ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE:** Destination file already exists and cannot be overwritten.

**ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION:** Copying face image file to destination file failed.

**ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND:** Extraction of the signature image failed – could not locate the signature rectangle.

**ID\_TRUE:** Function completed successfully.

### Remarks

Use this function to extract the signature image from the document image. The source document image is last scanned document (stored in the internal image buffer). If the parameter **sourceFile** is an empty string then the image is taken from the internal image buffer.

## GetIDSignatureDuplex

### Format

```
short GetIDSignatureDuplex(String sourceFile, String backSourceFile,
String destFile, short stateID)
```

### Parameters

szSourceFile – Full name of the source driver’s license image file.

szBackSourceFile – Full name of the source driver’s license image file second side.

szDestFile – Full name of the destination signature image.

stateID – The state id.

### Return value

**ID\_ERR\_FILE\_OPEN:** Cannot open input image file.

**INVALID\_INTERNAL\_IMAGE:** Invalid internal image file.

**ID\_FALSE:** Image processing failed.

**ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE:** Destination file already exists and cannot be overwritten.

**ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION:** Copying face image file to destination file failed.

**ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND:** Extraction of the face image failed – could not locate the face rectangle.

**ID\_TRUE:** Function completed successfully.

### Remarks

This function is for Duplex scan support that gives you the option to get the face image without knowing where it is (front or back side). Duplex scanning is available only with Scanshell 800DX\800DXN\3000D,5000.

Use this function to extract the face image from the document image. The source document image is last scanned document (stored in the internal image buffer). If the parameter **szSourceFile** is an empty string then the image is taken from the internal image buffer.

## GetStateNameById

### Format

```
String GetStateNameById(short stateID)
```

### Parameters

stateID – Constant value of the state.

### Return value

**ID\_ERR\_NO\_MATCH**: The input state id value does not match any state value.  
**State name.**

### Remarks

This function converts between the state id numeric value (constant short) and its textual name (string). The conversion can be from constant to string or from string to constant. The function examines the value of stateID. If this parameter is equal to -1 then the function takes the state name from the parameter szStateName and return the state id. If the string does not match any of the state names the function returns **ID\_ERR\_NO\_MATCH**. If the parameter stateID is not -1 then the function load the string szStateName with the state textual name and the function returns **ID\_TRUE**. If the stateID does not match any of the state constants then the function returns **ID\_ERR\_NO\_MATCH**.

The string compare are all case sensitive.

## GetIDStateShort

### Format

```
String GetIDStateShort(int stateID)
```

### Parameters

stateID – Constant value of the state.

### Return value

**ID\_ERR\_NO\_MATCH:** The input state id value does not match any state value.

**ID\_TRUE:** Found state constant that match.

**State name**

### Remarks

This function takes the stateID constant and load szStateName with the state abbreviated name. For example, if the input value is 0 (the constant value of the state Alabama) then the function loads szStateName with the string "AL".

## InitIdLib

### Format

```
short InitIdLib(String License)
```

### Parameters

szLicense – The library license key

### Return value

**LICENSE\_VALID:** License is valid and the library is ready to be used.

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**LICENSE\_EXPIRED:** License has expired. All scanner operations are disabled.

**LICENSE\_DOES\_NOT\_MATCH\_LIBRARY:** The license is invalid for this library. All library operations are disabled.

**GENERAL\_ERR\_PLUG\_NOT\_FOUND:** This error returns if either:

- The license is valid but no scanner is attached to the PC.
- The license is temporary but it has expired.

**SLIB\_LIBRARY\_ALREADY\_INITIALIZED:** The *InitLibrary* function call is ignored since the library is already loaded.

### Remarks

Use this function to initialize the IdCard library. This function must be called before calling any other function in the library.

## ProcessState

### Format

```
short ProcessState(String file, short stateID)
```

### Parameters

file – Empty strings  
stateID – Constant value of the state.

### Return value

#### Return

If function succeeds, the return value is **ID\_TRUE**.

If function fails, the return value is one of the following:

- **LICENSE\_INVALID** – Library was not initialized with proper license.
- **SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.
- **LICENSE\_INVALID** – The library was not initialized with the proper license.
- **SLIB\_ERR\_INVALID\_SCANNER** – No scanner was found attached to the PC.
- **ID\_ERR\_STATE\_NOT\_SUPPORTED** – The requested state id is not supported.
- **INVALID\_INTERNAL\_IMAGE** – No internal image is loaded. This value return when attempting to use this function without scanning an image first.

### Remarks

Use this function to process the internal image acquired in the last scan. The function de skew and cleans the image and then pass to the ocr for analysis. Processing the image does not modify the image content.

Successful image processing depends on the following:

- a. The image must be scanned in 24 bit (true color) and 300 dpi.
- b. The image must be aligned horizontally with tolerance of no more than  $\pm 10$  degrees.

## GetCntryNameById

### Format

```
String GetCntryNameById(short countyID)
```

### Parameters

**countyID** – Country ID as retrieved from the SDK

#### Return

**1) ID\_ERR\_NO\_MATCH:** The parameter *CountryName* does not contain known country name.

**2) Country name.**

## SetIDRegion

### Format

```
short SetIDRegion(int region)
```

### Parameters

**region** – The current region id.

### Return

**ID\_ERR\_AUTO\_DETECT\_NOT\_SUPPORTED:** The parameter *RegionId* does not support auto detection..

**ID\_ERR\_NO\_MATCH:** The parameter *RegionId* is not a valid region identifier.

**ID\_TRUE** – Region setting succeeded.

### Remarks

The SDK can automatically detect the card type using the following steps:

Setting the region number.

Running the function DetectState after scanning the image.

Currently, the regions that supports auto detections are:

USA (region number 0), Canada (region number 1), Australia (region number 4) and Asia (region number 5). The auto detection will support addition regions in the upcoming versions. For complete list of the regions and their related states please refer to appendix A.

**Note:** Upon invocation of the library, the default region is set to USA (region 0).

## SetIDRegionDetectionSequence

### Format

```
short SetIDRegionDetectionSequence(int region0, int region1, int region2, int region3, int region4, int region5, int region6)
```

### Parameters

[in] **RegionId0 - 6** – The region ids.

### Return

**ID\_ERR\_AUTO\_DETECT\_NOT\_SUPPORTED:** The parameter *RegionId* does not support auto detection..

**ID\_ERR\_NO\_MATCH:** The parameter *RegionId* is not a valid region identifier.

**ID\_TRUE** – Region setting succeeded.

### Remarks

Please read description of the function RegionSet.

This function will do the same but this time the sdk will try to find the card type in all the given regions id in this function.

Example: Call this function with 0 & 1 for USA & Canada and now when ever you call the function DetectStateEx with USA or Canada cards it will find the card.

## GetFirstIDRegion

### Format

```
short GetFirstIDRegion()
```

### Return

The value of the first region in the SDK region list.

### Remarks

The SDK divide the supported countries into several major regions. The regions are organized in a list that can be retrieved by calling to this function once, and then continuously calling the function *GetNextRegion* ().

Please notice the following:

The list size and order may vary in the future, as the SDK will include more and more regions. For example, regions 0 and 1 (USA and Canada respectively) are designated to be unified to a single region (North America region) in later versions. Therefore, you should never fix the region values in your code and always retrieve it using the function *GetFirstRegion* and *GetNextRegion*.

Not all the regions support auto detection. However, in the following versions more and more regions will support this feature.

## GetNextIDRegion

### Format

```
short GetNextIDRegion()
```

### Return

**ID\_ERR\_NO\_MATCH:** The function *GetFirstRegion* was never called prior to the call of this function.

**ID\_ERR\_NO\_NEXT\_COUNTRY:** The list has ended (the last region was retrieved in the previous function call).

Otherwise, the function returns the value of the next region id.

### Remarks

Use this function to retrieve the region list. To retrieve the list do the following:

Call the function *GetFirstRegion* is called once.

Call *GetNextRegion* continuously in a loop until the value

**ID\_ERR\_NO\_NEXT\_COUNTRY** is returned. For each call, the function returns the id of the next region.

## GetRegionNameById

### Format

```
String GetRegionNameById(short id)
```

### Parameters

**id** – The region numeric value.

### Return

- **ID\_ERR\_NO\_MATCH** – The parameter *id* is not a valid region enum
- **ID\_TRUE** – Region found. The *name* string returns loaded with the region name.
- **Region Name**

### Remarks

Use this helper function to convert region enum values into the region string name. This function, combined with the functions *GetFirstRegion* and *GetNextRegion* builds the region name list.

## GetRegionByCountry

### Format

```
short GetRegionByCountry(int countryId)
```



### Parameters

**countryId** – Constant value of a country

### Return value

**ID\_ERR\_NO\_MATCH** The parameter *CountryId* does not contain a valid country id. Otherwise, this function returns the region id that contains the input country.

### Remarks

Use this function to detect the region that contains the country. For complete list of the regions, countries and states please refer to appendix A.

## GetFirstCountryInRegion

### Format

```
short GetFirstCountryInRegion(int region)
```

### Parameters

**region** – The region id that contains the country list.

### Return

**ID\_ERR\_NO\_MATCH** – The value in *region* is not a valid region id.  
Country ID

Otherwise, the function returns the first country id in the region.

### Remarks

Use this function (combined with *GetNextCountryInRegion*) to obtain the list of countries in the region.

## GetNextCountryInRegion

### Format

```
short GetNextCountryInRegion(int region)
```

### Parameters

[in] **region** – The region id that contains the country list.

### Return

- **ID\_ERR\_NO\_MATCH** – The value in *region* is not a valid region id.
- **ID\_ERR\_NO\_NEXT\_COUNTRY** – There is no next country in the list – the last call retrieved the last country in the list.
- Otherwise, the function returns the next country id in the region.

### Remarks

Use this function (combined with *GetFirstCountryInRegion*) to obtain the list of countries in the region. You should call *GetFirstCountryInRegion* once, then call *GetNextCountryInRegion* continuously in a loop and store the returned value until the value **ID\_ERR\_NO\_NEXT\_COUNTRY** is returned. You can use the following strategy to build a complete region, country and state tree:

1. Retrieve the full region list using *GetFirstRegion* and *GetNextRegion* functions.
2. For each region, retrieve the full countries list using *GetFirstCountryInRegion* and *GetNextCountryInRegion* functions.
3. For each Country, retrieve the full state list using *GetFirstStateByCntry* and *GetNextStateByCntry* functions.

## GetIDFaceImgBuffer

### Format

```
String GetIDFacelmgBuffer(String source, String imageBufferType,
int nStateID)
```

### Parameters

**Source** – Full name of the source driver's license image file.

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

"BMP"

"JPG"

"PNG"

"TIF"

"TGA"

"PSD"

"PCX"

**StateID** – The state id

**Return**

Face Image Buffer

**Remarks**

Use this function to extract the face image from the document image to the given buffer. The source document image can be an existing image file (in BMP format) or the last scanned document (stored in the internal image buffer). If the parameter **SourceFile** is an empty string then the image is taken from the internal image buffer. If this parameter contains the full name of a valid image file then this file is used.

**GetIDSignatureImgBuffer**

**Format**

```
String GetIDSignatureImgBuffer(String source, String
imageBufferType, int nStateID)
```

**Parameters**

**Source** – Full name of the source driver’s license image file.

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

- “BMP”
- “JPG”
- “PNG”
- “TIF”
- “TGA”
- “PSD”
- “PCX”

**StateID** – The state id

**Return**

Signature Image Buffer

**Remarks**

Use this function to extract the signature image from the document image to the given buffer. The source document image can be an existing image file (in BMP format) or the last scanned document (stored in the internal image buffer). If the parameter **SourceFile** is an empty string then the image is taken from the internal image buffer. If this parameter contains the full name of a valid image file then this file is used.

## DetectProcessAndCompare

### Format

```
short DetectProcessAndCompare(String imageA, String imageB, int
stateID, int reserved)
```

### Parameters

**imageA** – Full name of the source driver’s license image file (side A).

**imageB** – Full name of the source driver’s license image file (Side B).

**state** – StateID can be -1 for auto detect state.

**reserved** – use the value 0.

### Return

If function succeeds, the return value is **the stateID**

If the function fails the value **-1** is returned.

### Remarks

Use this function to detect and process a card with two sides that each of the sides can contain text and/or barcode data. After using this function you can get the fields data from the CidData and Barcode classes (Using ScanShell 3000D\ScanShell800DX only).

## DetectProcessAndCompare2

### Format

```
short DetectProcessAndCompare2(String imageA, String imageB, int
stateID, int defaultBarCode, int reserved)
```

### Parameters

**imageA** – Full name of the source driver’s license image file (side A).

**imageB** – Full name of the source driver’s license image file (Side B).

**state** – StateID can be -1 for auto detect state.

**defaultBarCode-** This value will tell the function to use the 2D or 1D barcode as the first extraction process. 0=2D 1=1D

**reserved** – use the value 0.

### Return

If function succeeds, the return value is **the stateID**

If the function fails the value **-1** is returned.

### Remarks

Use this function to detect and process a card with two sides that each of the sides can contain text and/or barcode data. After using this function you can get the fields data from the CidData and Barcode classes (Using ScanShell 3000D\ScanShell800DX only).

### DetectProcessDuplex

#### Format

```
short DetectProcessDuplex (String imageA, String imageB, int stateID,
int reserved)
```

#### Parameters

**imageA** – Full name of the source driver’s license image file (side A).

**imageB** – Full name of the source driver’s license image file (Side B).

**state** – StateID can be -1 for auto detect state.

**reserved** – use the value 0.

#### Return

If function succeeds, the return value is **the stateID**

If the function fails the value **-1** is returned.

### Remarks

Use this function to detect the front side and to extract the data from the two sides of a card. After using this function you can get the fields value from the idData class (Using ScanShell 3000D\ ScanShell800DX only).

### ResetIDFields

#### Format

```
Void ResetIDFields ()
```

#### Parameters

None.

#### Return

None.

### Remarks

Use this function to clear the fields data from the IDData class.

## GetIDImageAngle

### Format

```
short GetIDImageAngle ()
```

### Parameters

None.

### Return

Angle of the image scanned. Should be called after DetectState function. Angle value can be used in RotateImage function to rotate the image.

## SetIDDatesFormat

### Format

```
void SetIDDatesFormat(short val)
```

### Parameters

[in] val – Needed date format type.

Can be one of these values: • EXTRACT\_DATE\_FORMAT\_NONE = 0 (Use the default date extraction) • EXTRACT\_DATE\_FORMAT\_MDY = 1 (mm-dd-yy) • EXTRACT\_DATE\_FORMAT\_DMY = 2 (dd-mm-yy) • EXTRACT\_DATE\_FORMAT\_YMD = 3 (yy-mm-dd) • EXTRACT\_DATE\_FORMAT\_YDM = 4 (yy-dd-mm)

### Return

Use this function to extract the date's fields with the format you need. This property will take effect on passport date fields as well.

## Data Functions

S. No.	Function Name	Return Value
1	getIDName	String

2	getIDFirstName	String
3	getIDMidName	String
4	getIDLastName	String
5	getIDName_L1	String
6	getIDName_L2	String
7	getIDNameSuffix	String
8	getID	String
9	getIDLicenseMain	String
10	getIDExpirationDate	String
11	getIDEyeColor	String
12	getIIDairColor	String
13	getIDEndorsements	String
14	getIDHeight	String
15	getIDWeight	String
16	getIDFee	String
17	getIDCounty	String
18	getIDAddress	String
19	getIDAddress2	String
20	getIDLAddress3	String
21	getIDLAddress4	String
22	getIDAddress5	String
23	getIDCity	String
24	getIDState	String
25	getIDCountry	String
26	getIDZip	String
27	getIDClass	String
28	getIDRestriction	String
29	getIDDateOfBirth	String
30	getIDSex	String
31	getIDSigNum	String
32	getIDType	String
33	getIDText1	String
34	getIDText2	String
35	getIDText3	String
36	getIDSide	String
37	getIDIssueDate	String
38	getIDSSNNumber	String
39	getIDDocType	String
40	getIDCountryShort	String
41	getIDDateOfBirthLong	String
42	getIDExpirationDateLong	String
43	getIDIssueDateLong	String

44	getIDNationality	String
45	getIDPlaceOfBirth	String
46	getIDPlaceOfIssue	String
47	getIDMotherName	String
48	getIDMotherName	String

## Barcode Library Functions

BarCode library functionality has a similar functionality as idData library. It extracts the data from 2D, PDF417 type bar code image. The library fetches the internal image (the last scanned image), process its graphic information and activates its internal image analyzer. The resultant text is kept in internal data structure ready to be retrieved by the application.

NOTE: The image MUST be scanned in 600dpi, gray scale (256 shades of gray) format.

### InitBCLib

#### Format

```
short InitBCLib(String license)
```

#### Parameters

license – The library license key

#### Return value

**LICENSE\_VALID:** License is valid and the library is ready to be used.

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**LICENSE\_EXPIRED:** License has expired. All scanner operations are disabled.

**LICENSE\_DOES\_NOT\_MATCH\_LIBRARY:** The license is invalid for this library. All library operations are disabled.

**GENERAL\_ERR\_PLUG\_NOT\_FOUND:** This error returns if either:

- The license is valid but no scanner is attached to the PC.
- The license is temporary but it has expired.

**SLIB\_LIBRARY\_ALREADY\_INITIALIZED:** The *InitBCLib* function call is ignored since the library is already loaded.

#### Remarks

Use this function to initialize the Barcode library. This function must be called before calling any other function in the library.



## ProcBarcode

### Format

short ProcessBarcode (String file, long reserved )

### Parameters

file – Empty string (“”)  
reserved1 – the value 0

If function succeeds, the return value is **BC\_ERR\_NONE**.

If function fails, the return value is one of the following:

- **LICENSE\_INVALID** – Library was not initialized with proper license.
- **SLIB\_ERR\_SCANNER\_NOT\_FOUND** – No attached scanner was found.
- **LICENSE\_INVALID** – The library was not initialized with the proper license.
- **SLIB\_ERR\_INVALID\_SCANNER** – No scanner was found attached to the PC.
- **ID\_ERR\_STATE\_NOT\_SUPORTED** – The requested state id is not supported.
- **INVALID\_INTERNAL\_IMAGE** – No internal image is loaded. This value return when attempting to use this function without scanning an image first.
- **BC\_ERR\_NO\_BC\_FOUND** – No barcode pattern (PDF417) was found on the image.

### Remarks

Use this function to process the internal image and extract 2D barcode information acquired in the last scan. The function de skew and cleans the image and then pass to the image analyzer for data extraction. The resultant textual data is kept in internal structure ready for retrieval by either *RefreshData()* or *GetRawText()* functions. Processing the image does not modify the image content.

Successful image processing depends on the following:

- a. The image must be scanned in 600 dpi – Gray scale color scheme.
- b. The image must be aligned in such way that the bar code image is vertical with tolerance of no more than  $\pm 10$  degrees.

## RefreshBC

### Format

short Refresh ()

### Parameters

None.

**Return Value**

If the function returns non-zero value, the data was retrieved successfully.  
 If the function returns zero value, the data was retrieved un-successfully.

**Remarks**

This function takes the raw data (obtained by *ProcessBC()*) and parse it according to the AAMVA standard.

**BC4DigitsYearFormat**

**Format**

short BC4YearDigitsFormat (short value)

**Parameters**

Value = 1 to get yyyy format of date.

**Return Value**

None

**Remarks**

This function is used to get dates in mm/dd/yyyy format.

**GetBCRawText**

**Format**

String GetBCRawText ()

**Parameters**

None

**Return Value**

Raw string data

**Remarks**

This function copies the raw data that was extracted from the barcode analyzer without further parsing. The buffer should be able to allocate up to 2048 characters. This function is useful to obtain data from general-purpose documents that use the PDF417 standard to export data.

## Data Functions

S. No.	Function Name	Return Value
1	getBCName	String
2	getBCFirstName	String
3	getBCMIDName	String
4	getBCLastName	String
5	getBCNameSuffix	String
6	getBCLicense	String
7	getBCDateOfBirth	String
8	getBCAddress	String
9	getBCCity	String
10	getBCState	String
11	getBCZip	String
12	getBCExpirationDate	String
13	getBCIssueDate	String
14	getBCSSN	String
15	getBCHeight	String
16	getBCWeight	String
17	getBCEyeColor	String
18	getBCHairColor	String

## Passport Library Functions

CPassport analyses and retrieve data from a standard passport image. The passport image is taken using ScanShell1000 scanner in either color or gray color scheme, analysed by the library and the result data is stored the library properties. The image may be a full image of the page (3"x5") or only the lower portion of the page (1"x5").

### InitPassportLib

#### Format

**short InitPassportLib (String license)**

#### Parameters

license – The library license key

### Return value

**LICENSE\_VALID:** License is valid and the library is ready to be used.

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**LICENSE\_EXPIRED:** License has expired. All scanner operations are disabled.

**LICENSE\_DOES\_NOT\_MATCH\_LIBRARY:** The license is invalid for this library. All library operations are disabled.

**GENERAL\_ERR\_PLUG\_NOT\_FOUND:** This error returns if the attached scanner is not one of the following scanners:

- ScanShell 600
- ScanShell 800
- ScanShell 1000

### Remarks

This function initializes the library. This function must be called before any other function in the library can be used.

### ProcessPassport

#### Format

```
short ProcessPassport(String file)
```

#### Parameters

file – "" Empty Strings.

#### Return value

**PASS\_ERR\_NONE** - If function succeeds, the return value is.

If function fails, the return value is one of the following:

**LICENSE\_INVALID** – The library was not initialized with the proper license.

**INVALID\_INTERNAL\_IMAGE** – No internal image is loaded. This value returns when attempting to use this function without scanning an image first

#### Remarks

Call this function to process the recently scanned passport image.

### ResetPassportData

#### Format

```
Void ResetPassportData ()
```

#### Parameters

None.

### Return

None.

### Remarks

Use this function to clear data from all the fields in the passport class.

## GetPassportMRZChecksumVerified

### Format

```
Int GetPassportMRZChecksumVerified (int FieldIndex)
```

### Parameters

FieldIndex – Constant value of the field.

### Return value

- **1 = Verification success.**
- **0 = Verification failed.**
- **-1 = Not been verified.**

### Remarks

Use this function to verify the field value with it's checksum digit on the MRZ line (where it was taken from).

## GetPassportFaceEx

### Format

```
short GetFacelImage(String sourceFile, String destFile)
```

### Parameters

sourceFile – Empty strings to use source file as buffer.

destFile – Null terminated string that holds the full name of the destination image file that will contain the face image from the passport.

### Return value

**PASS\_ERR\_NONE** - If function succeeds, the return value is.

If the function fails one of the following values is returned:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**PASS\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE**– Returned when a file with the same name as the destination file already exists and cannot be overwritten.

**PASS\_ERR\_CANNOT\_COPY\_TO\_DESTONATION**– Returned when the destination file cannot be opened for writes on the disk.

**PASS\_ERR\_FACE\_IMAGE\_NOT\_FOUND – Could not retrieve the face image from the passport image**

**Remarks**

Use this function to extract the image rectangle of the person’s face from the source passport image. Remember that the original scanned image must be rotated in 180 degrees (so it will be aligned correctly) before this function is called. This function works properly only for 3”x5” images.

**GetPassportFaceImgBuffer**

**Format**

**String GetPassportFaceImgBuffer (String source, String imageBufferType)**

**Parameters**

**Source** – Full name of the source passport image file. Use empty strings to use image buffer.

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

- “BMP”
- “JPG”
- “PNG”
- “TIF”
- “TGA”
- “PSD”
- “PCX”

**Return**

Passport face image buffer

**GetPassportSignature**

**Format**

**short GetPassportSignature(String sourceFile, String destFile)**

**Parameters**

sourceFile – Empty strings to use source file as buffer.

destFile – Null terminated string that holds the full name of the destination image file that will contain the signature image from the passport.

### Return value

**PASS\_ERR\_NONE** - If function succeeds, the return value is.

If the function fails one of the following values is returned:

**LICENSE\_INVALID** – Library was not initialized with proper license.

**PASS\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE**– Returned when a file with the same name as the destination file already exists and cannot be overwritten.

**PASS\_ERR\_CANNOT\_COPY\_TO\_DESTONATION**– Returned when the destination file cannot be opened for writes on the disk.

### Remarks

Use this function to extract the image rectangle of the person’s signature from the source passport image. Remember that the original scanned image must be rotated in 180 degrees (so it will be aligned correctly) before this function is called. This function works properly only for 3”x5” images.

## GetPassportSignatureImgBuffer

### Format

String GetPassportSignatureImgBuffer (String source, String imageBufferType)

### Parameters

**Source** – Full name of the source passport image file. Use empty strings to use image buffer.

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

“BMP”

“JPG”

“PNG”

“TIF”

“TGA”

“PSD”

“PCX”

### Return

Passport face signature buffer

## Data Functions

S.	Function Name	Return Value
----	---------------	--------------

No.		
1	getPassFirstName	String
2	getPassLastName	String
3	getPassMiddleName	String
4	getPassportNumber	String
5	getPassNationality	String
6	getPassNationalityLong	String
7	getPassDateOfBirth	String
8	getPassDateOfBirthLong	String
9	getPassSex	String
10	getPassExpirationDate	String
11	getPassExpirationDateLong	String
12	getPassPersonalNumber	String
13	getPassIssueDate	String
14	getPassIssueDateLong	String
15	getPassIssueCountryLong	String
16	getPassPlaceOfBirth	String

## ImageCtrl Library Functions

### InitImageLib

#### Format

```
short InitImageLib(String license)
```

#### Parameters

license – The library license key

#### Return value

**LICENSE\_VALID:** License is valid and the library is ready to be used.

**LICENSE\_INVALID:** The license is invalid. All scanner operations are disabled.

**LICENSE\_EXPIRED:** License has expired. All scanner operations are disabled.

**LICENSE\_DOES\_NOT\_MATCH\_LIBRARY:** The license is invalid for this library. All library operations are disabled.

**GENERAL\_ERR\_PLUG\_NOT\_FOUND:** This error returns if either:

- The license is valid but no scanner is attached to the PC.
- The license is temporary but it has expired.

#### Remarks

Use this function to initialize the CImageCtrl library. This function must be called before calling any other function in the library.



## GetImageColor

### Format

```
short GetImgColor(String fileName)
```

### Parameters

fileName – Image file name or empty string if evaluating the internal image.

### Return value

**IMG\_ERR\_FILE\_OPEN:** Cannot open input image file.

**INVALID\_INTERNAL\_IMAGE:** Internal image is invalid and cannot be analyzed.

**IMAGE\_BW** – The image has Black and White colors (1 bit image).

**IMAGE\_GRAY\_256** - The image has 256 colors of gray (8 bit image).

**IMAGE\_COLOR\_256** - The image has 256 colors (8 bit image).

**IMAGE\_COLOR\_TRUE** - The image has 16 million (24 bit image).

### Remarks

Use this function to obtain the image color scheme.

## RotateImage

### Format

```
short RotatelImage(String sourceImage, short angle, short destType, String destImage)
```

### Parameters

sourceImage – Full path name of the original image file. If this string is empty the rotation is performed on the internal image.

angle – The angle to rotate the original image. This value can be one of the following values:

- **ANGLE\_0** : 0 degrees rotation
- **ANGLE\_90**: 90 degrees rotation
- **ANGLE\_180**: 180 degrees rotation
- **ANGLE\_270**: 270 degrees rotation

destType- – The destination of the rotated image. This parameter may be one of two values:

- **SAVE\_TO\_FILE** : Save the image to a file. The file name should be given in *DestImage* parameter.
- **SAVE\_TO\_CLIPBOARD** : Copy the rotated image the image to the clipboard.

*destImage* - Full path name of the destination file. This parameter is ignored if the parameter *destType* is set to **SAVE\_TO\_CLIPBOARD**. If this value is an empty string no save will be performed

### Return value

If function succeeds, the return the value **IMG\_ERR\_SUCCESS**.

If the function fails it returns one of the following values:

- **LICENSE\_INVALID** – Library was not initialized with proper license.
- **IMG\_ERR\_BAD\_ANGLE\_0** – bad rotation parameter.
- **IMG\_ERR\_BAD\_DESTINATION** – Bad destination parameter (the destination parameter is neither file or clipboard)
- **IMG\_ERR\_FILE\_OPEN** – Cannot open input file. This value is returned if the *SourceImage* string is not empty but it points to a missing or invalid source image file.
- **INVALID\_INTERNAL\_IMAGE** – This value is returned if the *SourceImage* string is empty but no document was scanned so there is no internal image in the memory.
- **IMG\_ERR\_FILE\_SAVE\_TO\_CLIPBOARD** – Cannot save image to clipboard due to an error.
- **IMG\_ERR\_FILE\_SAVE\_TO\_FILE** – Cannot save destination file due to invalid destination file or disk save error

### Remarks

Use this function to rotate an image in 0, 90, 180 or 270 degrees and save it to a file in any one of seven formats. The manipulated image may be loaded from an external file (if *src* string holds a string value equal to the source image file name) or performed on the internal image buffer (if *src* string is empty). When using a file as the image source, it is important to use the proper file extension to indicate the image format. Proper extension types are: BMP, JPG, TIFF, PCX, PNG, TGA and PSD. If an image has unrecognizable extension due to an error (e.g. TIFF instead of TIF) the function refers to the file as BITMAP.

After the image is rotated it can be exported to either the clipboard or to external image file. The destination file name may be one of the seven file formats indicated above. If an image has unrecognizable extension due to an error (e.g. TIFF instead of TIF) the function exports to the file in a BITMAP format. The destination file name may be the same as the source file name. In such case the new file, resulting with a rotated image, will overwrite the original file. If no destination image file name is given (empty string) no save is done.

Do not be misled by the name of this function. This function flexibility actually allows implicitly to do the following:

- Use the following function call to convert an image file from one type to another:  
**RotatImage ("xxx.bmp", ANGLE\_0, SAVE\_TO\_FILE, "xxx.jpg")**
- Use the following function call to copy an image file to the clipboard:  
**RotatImage ("xxx.bmp", ANGLE\_0, SAVE\_TO\_CLIPBOARD, "")**
- Use the following function call to rotate the internal image :  
**RotatImage ("", ANGLE\_0, SAVE\_TO\_FILE, "")**
- Use the following function call to save the internal image to a file:  
**RotatImage ("", ANGLE\_0, SAVE\_TO\_FILE, "xxx.bmp")**

## ReformatImage

### Format

```
short ReformatImage(String sourceImage, short toColor, short toDpi,
String destImage)
```

### Parameters

sourceImage – Full path name of the original image file. If this string is empty the rotation is performed on the internal image.

toColor – One of five values:

- **LICENSE\_INVALID** – Library was not initialized with proper license.
- **IMAGE\_SAME\_COLOR** – no modification in the image color scheme
- **IMAGE\_BW** – Convert to black and white color scheme.
- **IMAGE\_GRAY\_256** – convert to 256 gray scale color scheme.
- **IMAGE\_COLOR\_256** – convert to 256-color scheme.
- **IMAGE\_COLOR\_TRUE** – convert to true color scheme

toDpi – Set the new Image DPI. A value of 0 indicate no DPI modification

destImage – Full path name of the destination file. Is this value is an empty string no save will be performed

### Return value

If function succeeds, the return the value **IMG\_ERR\_SUCCESS**.

If the function fails it returns one of the following values:

- **IMG\_ERR\_BAD\_COLOR** – Bad toColor parameter value.
- **IMG\_ERR\_BAD\_DPI** – Bad toDpi parameter value.
- **IMG\_ERR\_FILE\_OPEN** – Cannot open input file. This value is returned if the src string is not empty but it points to a missing or invalid source image file.

- **INVALID\_INTERNAL\_IMAGE** – This value is returned if the src string is empty but no document was scanned so there is no internal image in the memory.
- **IMG\_ERR\_FILE\_SAVE\_TO\_FILE** – Cannot save destination file.
- **IMG\_ERR\_FILE\_SAVE\_TO\_FILE** – Cannot save destination file due to invalid destination file or disk save error

**Remarks**

Use this function to modify the image color scheme and resolution and save it to a file in any one of seven formats. The manipulated image may be loaded from an external file (if src string holds a string value equal to the source image file name) or performed on the internal image buffer (if src string is empty). When using a file as the image source, it is important to use the proper file extension to indicate the image format. Proper extension types are: BMP, JPG, TIFF, PCX, PNG, TGA and PSD. If an image has unrecognizable extension due to an error (e.g. TIFF instead of TIF) the function refers to the file as BITMAP.

Image reformat can be done either on the image color scheme or the image dpi or both. Notice that changing the image format may lose the image color information (e.g., when converting from 24 bit true color to 256 gray scale).

Modifying and image format from 256 gray scales to 24 bit true color will (obviously) not add color to the image but it will save the image with the proper RGB format (no color map) instead of or using 256 gray scale palette.

After the image is reformatted it can be exported to external image file. The destination file name may be one of the seven file formats indicated above. If the destination file name has unrecognizable extension the function exports to the file in a BITMAP format (the default format). If no destination image file name is given (empty string) no save is done.

**Important: The 256 colors scheme is NOT supported for JPG and TIF files**

Destination image extension	Destination image type			
	True color (24 bit)	256 colors (8 bit)	Gray scale (8 bit)	Black and white (1 bit)
BMP	√	√	√	√
TIF	√		√	√
JPG	√		√	
PCX	√	√	√	
TGA	√	√	√	
PNG	√	√	√	
PSD	√			

**GetImageBufferData**

**Format**

```
String GetImageBufferData(String imageBufferType)
```

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

"BMP"  
 "JPG"  
 "PNG"  
 "TIF"  
 "TGA"  
 "PSD"  
 "PCX"

**Return**

Image Buffer

**GetImageBufferBackData**

**Format**

```
String GetImageBufferBackData(String imageBufferType)
```

**imageBufferType** – Set the buffer image format. May be one of the following strings (case insensitive):

"BMP"  
 "JPG"  
 "PNG"  
 "TIF"  
 "TGA"  
 "PSD"  
 "PCX"

**Return**

Image Buffer of the back side of the card

**SetMainImage**

**Format**

```
SetMainImage (iImageType As Integer) As Integer
```

## Parameters

**iImageType:** Set the main image buffer index. Available values are:

MAIN\_IMAGE (0): The front side image from the scanner.

MAIN\_BC\_IMAGE (1): The front side barcode image (for SnapShell model only).

BACK\_IMAGE (2): The Back side image from the scanner

BACK\_BC\_IMAGE (3): The Back side barcode image from the scanner (for SnapShell model only).

## Return

**IMG\_ERR\_BAD\_PARAM-** return if iImageType is not one of the values 0-3.

**IMG\_ERR\_SUCCESS** – Function processed successfully.

## Remarks

Use this function to select the current image that the SDK will use as main image. This allows to do image manipulation on the back image. In such case you need to make sure to switch back to the main image once the back image manipulation is done.

## Appendix A: Definitions

The following values are used as constants:

### 1. Library SlibEx constants

' Scanner color scheme types

Public Const GRAY = 1

Public Const BW = 2

Public Const HT = 3

Public Const TRUECOLOR = 4

' Scanner return values

Public Const SLIB\_FALSE = 0

Public Const SLIB\_TRUE = 1

' Scanner general error types

Public Const SLIB\_ERR\_NONE = 1

Public Const SLIB\_ERR\_INVALID\_SCANNER = -1

' Scanning failure definition

Public Const SLIB\_ERR\_SCANNER\_GENERAL\_FAIL = -2

Public Const SLIB\_ERR\_CANCELED\_BY\_USER = -3

Public Const SLIB\_ERR\_SCANNER\_NOT\_FOUND = -4

Public Const SLIB\_ERR\_HARDWARE\_ERROR = -5

Public Const SLIB\_ERR\_PAPER\_FED\_ERROR = -6

Public Const SLIB\_ERR\_SCANABORT = -7

Public Const SLIB\_ERR\_NO\_PAPER = -8

```

Public Const SLIB_ERR_PAPER_JAM = -9
Public Const SLIB_ERR_FILE_IO_ERROR = -10
Public Const SLIB_ERR_PRINTER_PORT_USED = -11
Public Const SLIB_ERR_OUT_OF_MEMORY = -12

Public Const SLIB_ERR_BAD_WIDTH_PARAM = -2
Public Const SLIB_ERR_BAD_HEIGHT_PARAM = -3

Public Const SLIB_ERR_BAD_PARAM = -2

Public Const SLIB_LIBRARY_ALREADY_INITIALIZED = -13
Public Const SLIB_ERR_DRIVER_NOT_FOUND = -14
Public Const SLIB_ERR_SCANNER_BUSSY = -15
Public Const SLIB_ERR_IMAGE_CONVERSION = -16
Public Const SLIB_UNLOAD_FAILED_BAD_PARENT = -17
Public Const SLIB_NOT_INITIALIZED = -18
Public Const SLIB_LIBRARY_ALREADY_USED_BY_OTHER_APP = -19
Public Const SLIB_CONFLICT_WITH_INOUTSCAN_PARAM = -20
Public Const SLIB_CONFLICT_WITH_SCAN_SIZE_PARAM = -21

' Button definition for ScanShell1000
Public Const TOP_BUTTON = 1
Public Const MIDDLE_BUTTON = 3
Public Const BOTTOM_BUTTON = 2

'Error values for multiple devices management
Public Const SLIB_NO_SUPPORT_MULTIPLE_DEVICES = -22
Public Const SLIB_ERR_CAM_ALREADY_ASSIGNED = -23
Public Const SLIB_ERR_NO_FREE_CAM_FOUND = -24
Public Const SLIB_ERR_CAM_NOT_FOUND = -25
Public Const SLIB_ERR_CAM_NOT_ASSIGNED_TO_THIS_APP = -26
Public Const GENERAL_ERR_PLUG_NOT_FOUND = -200
Public Const ERR_SCANNER_ALREADY_IN_USE = -201
Public Const SLIB_ERR_SCANNER_ALREADY_IN_USE = -202
Public Const SLIB_ERR_CANNOT_OPEN_TWAIN_SOURCE = -203
Public Const SLIB_ERR_NO_TWAIN_INSTALLED = -204
Public Const SLIB_ERR_NO_NEXT_VALUE = -205

```

## 2. Library idData Constants

```

' Country definitions
Public Const COUNTRY_USA = 0
Public Const COUNTRY_AUSTRALIA = 1
Public Const COUNTRY_MALAYSIA = 2
Public Const COUNTRY_CANADA = 3
Public Const COUNTRY_CHILE = 4

```

Public Const COUNTRY\_FRANCE = 5  
 Public Const COUNTRY\_MEXICO = 6  
 Public Const COUNTRY\_UNITED\_KINGDOM = 7  
 Public Const COUNTRY\_BRAZIL = 8  
 Public Const COUNTRY\_ISRAEL = 9  
 Public Const COUNTRY\_GERMANY = 10  
 Public Const COUNTRY\_SPAIN = 11  
 Public Const COUNTRY\_ROMANIA = 12  
 Public Const COUNTRY\_BERMUDA = 13  
 Public Const COUNTRY\_SINGAPORE = 14  
 Public Const COUNTRY\_NORWAY = 15  
 Public Const COUNTRY\_NEW\_ZELAND = 16  
 Public Const COUNTRY\_HOLAND = 17  
 Public Const COUNTRY\_LUX = 18  
 Public Const COUNTRY\_LITHUANIA = 19  
 Public Const COUNTRY\_SWISS = 20  
 Public Const COUNTRY\_BAHAMAS = 21  
 Public Const COUNTRY\_SWEDEN = 22  
 Public Const COUNTRY\_ITALY = 23  
 Public Const UNIVERSITY\_USA = 24  
 Public Const COUNTRY\_TURKEY = 25  
 Public Const EMPLOYMENT\_CARDS = 26  
 Public Const COUNTRY\_POLAND = 27  
 Public Const COUNTRY\_PERU = 29  
 Public Const COUNTRY\_PUERTO\_RICO = 30  
 Public Const COUNTRY\_PORTUGAL = 31  
 Public Const COUNTRY\_NICARAGUA = 32  
 Public Const COUNTRY\_GUATEMALA = 33  
 Public Const COUNTRY\_EL\_SALVADOR = 34  
 Public Const COUNTRY\_SOUTH\_AFRICA = 35  
 Public Const COUNTRY\_PANAMA = 36  
 Public Const COUNTRY\_INDONESIA = 37  
 Public Const COUNTRY\_BELGIUM = 38  
 Public Const SERVICE\_CARDS = 39  
 Public Const ENTERTAINMENT\_CARDS = 40  
 Public Const COUNTRY\_CROATIA = 41  
 Public Const USAPILOTS\_CARDS = 42  
 Public Const COUNTRY\_CHINA = 43  
 Public Const ACCESS\_CARDS = 44  
 Public Const COUNTRY\_BULGARIA = 45  
 Public Const fine EUROPE\_GENERAL\_CARDS = 46  
 Public Const COUNTRY\_CZECH = 47  
 Public Const COUNTRY\_BOSNIA = 48  
 Public Const COUNTRY\_HUNGARY = 49  
 Public Const COUNTRY\_SLOVAKIA = 50  
 Public Const COUNTRY\_KOSOVO = 51  
 Public Const OCB\_CARDS = 52  
 Public Const COUNTRY\_SLOVENIA = 53



Public Const COUNTRY\_IRELAND = 54  
 Public Const COUNTRY\_UAE = 55  
 Public Const COUNTRY\_BAHRAIN = 56  
 Public Const COUNTRY\_AUSTRIA = 57  
 Public Const COUNTRY\_RUSSIA = 58  
 Public Const COUNTRY\_SERBIA = 59  
 Public Const COUNTRY\_BOLIVIA = 60  
 Public Const SPAIN\_POLICE\_CARDS = 61  
 Public Const COUNTRY\_LIECHTENSTEIN = 63  
 Public Const COUNTRY\_FINLAND = 64  
 Public Const EHIC\_CARDS = 65  
 Public Const COUNTRY\_ECUADOR = 67  
 Public Const COUNTRY\_BRUNEI = 68  
 Public Const COUNTRY\_HONDURA = 69  
 Public Const SCISIUSAC\_CARDS = 70  
 Public Const COUNTRY\_ESTONIA = 71  
 Public Const COUNTRY\_DENMARK = 72  
 Public Const COUNTRY\_DOMINICAN\_REPUBLIC = 73  
 Public Const COUNTRY\_HAITI = 74  
 Public Const USAA\_CARDS = 75  
 Public Const COUNTRY\_CYPRUS = 76  
 Public Const AMPORT\_CARDS = 77  
 Public Const COUNTRY\_ISLAND = 78  
 Public Const COUNTRY\_COLUMBIA = 79  
 Public Const COUNTRY\_VENEZUELA = 80  
 Public Const COUNTRY\_INDIA = 81  
 Public Const COUNTRY\_NAMIBIA = 82  
 Public Const COUNTRY\_ZAMBIA = 83  
 Public Const PH\_CARDS\_CARDS = 84  
 Public Const COUNTRY\_OMAN = 85  
 Public Const COUNTRY\_QATAR = 86  
 Public Const COUNTRY\_SAUDI\_ARABIA = 87  
 Public Const COUNTRY\_ANDORRA = 88  
 Public Const COUNTRY\_GUERNSEY = 89  
 Public Const COUNTRY\_ISLE\_OF\_MAN = 90  
 Public Const COUNTRY\_LATVIA = 91  
 Public Const COUNTRY\_MALTA = 92  
 Public Const COUNTRY\_ARGENTINA = 93  
 Public Const COUNTRY\_ST\_CHRIST\_NEVIS = 94  
 Public Const COUNTRY\_ALBANIA = 95  
 Public Const IRELAND\_FIREARM\_CARDS = 96  
 Public Const COUNTRY\_MONTENEGRO = 97  
 Public Const COUNTRY\_KENYA = 98  
 Public Const COUNTRY\_NIGERIA = 99  
 Public Const COUNTRY\_MACEDONIA = 100  
 Public Const COUNTRY\_MOROCCO = 101  
 Public Const COUNTRY\_PHILIPPINES = 102  
 Public Const COUNTRY\_TURKS\_CAICOS = 103

Public Const COUNTRY\_THAILAND = 104  
Public Const COUNTRY\_MOLDOVA = 105

' states definitions

Public Const ID\_ALABAMA = 0  
Public Const ID\_ALASKA = 1  
Public Const ID\_ARIZONA = 2  
Public Const ID\_ARKANSAS = 3  
Public Const ID\_CALIFORNIA = 4  
Public Const ID\_COLORADO = 5  
Public Const ID\_CONNECTICUT = 6  
Public Const ID\_DELAWARE = 7  
Public Const ID\_WASHINGTON\_DC = 8  
Public Const ID\_FLORIDA = 9  
Public Const ID\_GEORGIA = 10  
Public Const ID\_IDAHO = 11  
Public Const ID\_ILLINOIS = 12  
Public Const ID\_INDIANA = 13  
Public Const ID\_IOWA = 14  
Public Const ID\_KANSAS = 15  
Public Const ID\_KENTUCKY = 16  
Public Const ID\_LOUISIANA = 17  
Public Const ID\_MAINE = 18  
Public Const ID\_MARYLAND = 19  
Public Const ID\_MASSACHUSETTS = 20  
Public Const ID\_MICHIGAN = 21  
Public Const ID\_MINNESOTA = 22  
Public Const ID\_MISSISSIPPI = 23  
Public Const ID\_MISSOURI = 24  
Public Const ID\_MONTANA = 25  
Public Const ID\_NEBRASKA = 26  
Public Const ID\_NEVADA = 27  
Public Const ID\_NEW\_HAMPSHIRE = 28  
Public Const ID\_NEW\_JERSEY = 29  
Public Const ID\_NEW\_MEXICO = 30  
Public Const ID\_NEW\_YORK = 31  
Public Const ID\_NORTH\_CAROLINA = 32  
Public Const ID\_NORTH\_DAKOTA = 33  
Public Const ID\_OHIO = 34  
Public Const ID\_OKLAHOMA = 35  
Public Const ID\_OREGON = 36  
Public Const ID\_PENNSYLVANIA = 37  
Public Const ID\_RHODE\_ISLAND = 38  
Public Const ID\_SOUTH\_CAROLINA = 39  
Public Const ID\_SOUTH\_DAKOTA = 40

Public Const ID\_TENNESSEE = 41  
Public Const ID\_TEXAS = 42  
Public Const ID\_UTAH = 43  
Public Const ID\_VERMONT = 44  
Public Const ID\_VIRGINIA = 45  
Public Const ID\_WASHINGTON = 46  
Public Const ID\_WEST\_VIRGINIA = 47  
Public Const ID\_WISCONSIN = 48  
Public Const ID\_WYOMING = 49  
Public Const ID\_HAWAII = 54  
Public Const ID\_GREEN\_CARD = 81  
Public Const ID\_ARMY\_CARD = 82  
Public Const ID\_SSN\_CARD = 83  
Public Const ID\_NYPD = 84  
Public Const ID\_MEXICO\_USA = 85  
Public Const ID\_GUAM = 86  
Public Const ID\_CEMA\_COMPLIANT = 87

Public Const ID\_NSW = 50  
Public Const ID\_ACT = 51  
Public Const ID\_QLD = 52  
Public Const ID\_VIC = 53  
Public Const ID\_TAS = 55  
Public Const ID\_WST = 56  
Public Const ID\_SA = 57  
Public Const ID\_NT = 58  
Public Const ID\_COOK\_ISLANDS = 59

Public Const ID\_MALAYSIA = 60  
Public Const ID\_SINGAPORE = 180  
Public Const ID\_NEW\_ZELAND = 200

Public Const ID\_ONTARIO = 70  
Public Const ID\_ALBERTA = 71  
Public Const ID\_BRITISH\_COLUMBIA = 72  
Public Const ID\_MANITOBA = 73  
Public Const ID\_NEW\_BRUNSWICK = 74  
Public Const ID\_NEW\_FOUNDLAND = 75  
Public Const ID\_NWTERITORIES = 76  
Public Const ID\_NOVASCOTIA = 77  
Public Const ID\_SASKATCHEWAN = 78  
Public Const ID\_CANADA\_CITIZEN\_ID = 79  
Public Const ID\_QUEBEC = 1079

Public Const ID\_CHILE = 80  
Public Const ID\_FRANCE = 90  
Public Const ID\_MEXICO = 100  
Public Const ID\_UNITED\_KINGDOM = 110

Public Const ID\_ISRAEL = 120  
Public Const ID\_BRAZIL = 130  
Public Const ID\_GERMAN\_ID = 140  
Public Const ID\_GERMAN\_LIC = 141  
Public Const ID\_SPAIN = 150  
Public Const ID\_ROMANIA = 160  
Public Const ID\_NORWAY = 190  
Public Const ID\_HOLAND = 210  
Public Const ID\_LUX = 220  
Public Const ID\_LITHUANIA = 230  
Public Const ID\_SWISS = 240  
Public Const ID\_SWEDEN = 260  
Public Const ID\_ITALY = 270  
Public Const ID\_UNIVERSITY\_USA = 280  
Public Const ID\_TURKEY = 290  
Public Const ID\_EMPLOY = 300  
Public Const ID\_POLAND = 310  
Public Const ID\_COSTA\_RICA = 320  
Public Const ID\_PERU = 330  
Public Const ID\_PUERTO\_RICO = 340  
Public Const ID\_PORTUGAL = 350  
Public Const ID\_NICARAGUA = 360  
Public Const ID\_GUATEMALA = 370  
Public Const ID\_EL\_SALVADOR = 380  
Public Const ID\_SOUTH\_AFRICA = 390  
Public Const ID\_PANAMA = 400  
Public Const ID\_INDONESIA = 410  
Public Const ID\_BELGIUM = 420  
Public Const ID\_PROTECTIVE = 430  
Public Const ID\_ENTERTAINMENT = 440  
Public Const ID\_CROATIA = 450  
Public Const ID\_USAPILOTS = 460  
Public Const ID\_CHINA = 470  
Public Const ID\_ACCESS = 480  
Public Const ID\_BULGARIA = 490  
Public Const ID\_KEYPASS = 500  
Public Const ID\_EUROPE\_GENERAL\_CARDS = 510  
Public Const ID\_CZECH = 520  
Public Const ID\_BOSNIA = 530  
Public Const ID\_HUNGARY = 540  
Public Const ID\_SLOVAKIA = 550  
Public Const ID\_KOSOVO = 560  
Public Const OCB\_CARDS = 570  
Public Const ID\_SLOVENIA = 580  
Public Const ID\_MRZ\_INFO = 590  
Public Const ID\_IRELAND = 600  
Public Const ID\_UAE = 610  
Public Const ID\_BAHRAIN = 620

Public Const ID\_SPAIN\_POLICE = 630  
 Public Const ID\_AUSTRIA = 640  
 Public Const ID\_RUSSIA = 650  
 Public Const ID\_SERBIA = 660  
 Public Const ID\_BOLIVIA = 670  
 Public Const ID\_LIECHTENSTEIN = 680  
 Public Const ID\_FINLAND = 690  
 Public Const ID\_EHIC = 700  
 Public Const ID\_ECUADOR = 710  
 Public Const ID\_BRUNEI = 720  
 Public Const ID\_HONDURAS = 730  
 Public Const ID\_SCSIUSAC = 740  
 Public Const ID\_ESTONIA = 750  
 Public Const ID\_DENMARK = 760  
 Public Const ID\_DOMINICAN\_REPUBLIC = 770  
 Public Const ID\_HAITI = 780  
 Public Const ID\_USAA = 790  
 Public Const ID\_CYPRUS = 800  
 Public Const ID\_AMPORT = 810  
 Public Const ID\_ISLAND = 820  
 Public Const ID\_COLUMBIA = 830  
 Public Const ID\_VENEZUELA = 840  
 Public Const ID\_INDIA = 850  
 Public Const ID\_NAMIBIA = 860  
 Public Const ID\_ZAMBIA = 870  
 Public Const ID\_PRINCEEDWARD = 880  
 Public Const ID\_PH\_CARDS = 890  
 Public Const ID\_OMAN = 900  
 Public Const ID\_QATAR = 910  
 Public Const ID\_POSITIVE\_ACCESS\_CARDS = 920  
 Public Const ID\_SAUDI\_ARABIA = 930  
 Public Const ID\_ANDORRA = 940  
 Public Const ID\_GUERNSEY = 950  
 Public Const ID\_ISLE\_OF\_MAN = 960  
 Public Const ID\_LATVIA = 970  
 Public Const ID\_MALTA = 980  
 Public Const ID\_ARGENTINA = 990  
 Public Const ID\_ST\_CHRIST\_NEVIS = 1000  
 Public Const ID\_ALBANIA = 1010  
 Public Const ID\_IRELAND\_FIREARM = 1020  
 Public Const ID\_MONTENEGRO = 1030  
 Public Const ID\_KENYA = 1040  
 Public Const ID\_NIGERIA = 1050  
 Public Const ID\_MACEDONIA = 1060  
 Public Const ID\_MOROCCO = 1070  
 Public Const ID\_PHILIPPINES = 1080  
 Public Const ID\_TURKS\_CAICOS = 1090  
 Public Const ID\_THAILAND = 1100

Public Const ID\_MOLDOVA = 1110

' region definitions

Public Const REGION\_USA = 0  
 Public Const REGION\_CANADA = 1  
 Public Const REGION\_SOUTH\_AMERICA = 2  
 Public Const REGION\_EUROPE = 3  
 Public Const REGION\_AUSTRALIA = 4  
 Public Const REGION\_ASIA = 5  
 Public Const REGION\_GENERAL\_DOC = 6  
 Public Const REGION\_AFRICA = 7

Public Const FRONT\_IMAGE\_SIDE = 0  
 Public Const BACK\_IMAGE\_SIDE = 1  
 Public Const UNKNOWN\_IMAGE = 2

'Date formats

Public Const EXTRACT_DATE_FORMAT_NONE = 0	'Use defaults
Public Const EXTRACT_DATE_FORMAT_MDY = 1	'mm-dd-yy
Public Const EXTRACT_DATE_FORMAT_DMY = 2	'dd-mm-yy
Public Const EXTRACT_DATE_FORMAT_YMD = 3	'yy-mm-dd
Public Const EXTRACT_DATE_FORMAT_YDM = 4	'yy-dd-mm

' function return values

Public Const ID\_TRUE = 1  
 Public Const ID\_FALSE = 0

' error enums

Public Const ID\_ERR\_NONE = 1  
 Public Const ID\_ERR\_STATE\_NOT\_SUPORTED = -2  
 Public Const ID\_ERR\_BAD\_PARAM = -3  
 Public Const ID\_ERR\_NO\_MATCH = -4  
 Public Const ID\_ERR\_FILE\_OPEN = -5  
 Public Const ID\_BAD\_DESTINATION\_FILE = -6  
 Public Const ID\_ERR\_FEATURE\_NOT\_SUPPORTED = -7

Public Const ID\_ERR\_COUNTRY\_NOT\_INIT = -8  
 Public Const ID\_ERR\_NO\_NEXT\_COUNTRY = -9  
 Public Const ID\_ERR\_STATE\_NOT\_INIT = -10  
 Public Const ID\_ERR\_NO\_NEXT\_STATE = -11  
 Public Const ID\_ERR\_CANNOT\_DELETE\_DESTINATION\_IMAGE = -12  
 Public Const ID\_ERR\_CANNOT\_COPY\_TO\_DESTONATION = -13  
 Public Const ID\_ERR\_FACE\_IMAGE\_NOT\_FOUND = -14

' state auto-detect error

Public Const ID\_ERR\_STATE\_NOT\_RECOGNIZED = -15  
 Public Const ID\_ERR\_USA\_TEMPLATES\_NOT\_FOUND = -16  
 Public Const ID\_ERR\_WRONG\_TEMPLATE\_FILE = -17

Public Const ID\_ERR\_REGION\_NOT\_INIT = -18  
Public Const ID\_ERR\_AUTO\_DETECT\_NOT\_SUPPORTED = -19  
Public Const ID\_ERR\_COMPARE\_NO\_TEXT= -20  
Public Const ID\_ERR\_COMPARE\_NO\_BARCODE = -21  
Public Const ID\_ERR\_COMPARE\_BC\_LIB\_NOT\_FOUND = -22  
Public Const ID\_ERR\_COMPARE\_LICENSE\_DONT\_MATCH\_BC\_LIBRARY= -23  
Public Const ID\_ERR\_DM\_LIBRARY\_NOT\_FOUND= -24  
Public Const ID\_ERR\_DM\_LIBRARY\_NOT\_LOADED= -25  
Public Const ID\_ERR\_DM\_WM\_NOT\_FOUND= -26  
Public Const ID\_ERR\_DM\_WM\_NOT\_AUTHENTICATED = -27

`GetFaceEx imageType values (only at supported cards see function description)

Public Const FACE\_IMAGE\_TYPE\_REGULAR = 0  
Public Const FACE\_IMAGE\_TYPE\_WITH\_FRAME = 1

### 3. Library CImage constants

' return values

```
Public Const IMG_ERR_SUCCESS = 0
Public Const IMG_ERR_FILE_OPEN = -100
Public Const IMG_ERR_BAD_ANGLE_0 = -101
Public Const IMG_ERR_BAD_ANGLE_1 = -102
Public Const IMG_ERR_BAD_DESTINATION = -103
Public Const IMG_ERR_FILE_SAVE_TO_FILE = -104
Public Const IMG_ERR_FILE_SAVE_TO_CLIPBOARD = -105
Public Const IMG_ERR_FILE_OPEN_FIRST = -106
Public Const IMG_ERR_FILE_OPEN_SECOND = -107
Public Const IMG_ERR_COMB_TYPE = -108
```

```
Public Const IMG_ERR_BAD_COLOR = -130
Public Const IMG_ERR_BAD_DPI = -131
Public Const INVALID_INTERNAL_IMAGE = -132
```

' image saving target definition

```
Public Const SAVE_TO_FILE = 0
Public Const SAVE_TO_CLIPBOARD = 1
```

' image rotation angle definitions

```
Public Const ANGLE_0 = 0
Public Const ANGLE_90 = 1
Public Const ANGLE_180 = 2
Public Const ANGLE_270 = 3
```

' image combination options

```
Public Const IMAGE_COMB_VERTICAL = 0
Public Const IMAGE_COMB_HORIZONTAL = 1
```

' image color conversion

```
Public Const IMAGE_SAME_COLOR = 0
Public Const IMAGE_BW = 1
Public Const IMAGE_GRAY_256 = 2
Public Const IMAGE_COLOR_256 = 3
Public Const IMAGE_COLOR_TRUE = 4
```

'text stamp positions

```
Public Const IMAGE_TOP = 0
Public Const IMAGE_MIDDLE = 1
Public Const IMAGE_BOTTOM = 2
```

```
Public Const IMAGE_LEFT = 0
```



```
Public Const IMAGE_MID_HOR = 1
Public Const IMAGE_RIGHT = 2
```

## 4. Library COcr constants

```
' return values
Public Const TOCR_SUCCESS = 1
Public Const TOCRJOBERROR = -2
Public Const TOCR_BAD_TYPE = -3

' OCR text type detection
Public Const USE_ALPHANUM = 0
Public Const USED_NUM_ONLY = 2
Public Const USE_ALPHA_CAPS_ONLY = 3
```

## 5. License related constants

```
Public Const LICENSE_VALID = 1
Public Const LICENSE_EXPIRED = -20
Public Const LICENSE_INVALID = -21
Public Const LICENSE_DOES_NOT_MATCH_LIBRARY = -22
Public Const GENERAL_ERR_PLUG_NOT_FOUND = -200
```

## 6. Library CBarcode constants

```
' List of all driver license field indexes
' The values of the fields is extracted directly from the bar code analyzer.
Public Const BCF_NAME = 0
Public Const BCF_ADDRESS = 1
Public Const BCF_CITY = 2
Public Const BCF_JURISTRDICTION_CODE = 3
Public Const BCF_POASTAL_CODE = 4
Public Const BCF_LICENSE_MAIN = 5
Public Const BCF_CLASS = 6
Public Const BCF_RESTRICTION = 7
Public Const BCF_ENDORSEMENT = 8
Public Const BCF_EXPIRES = 9
Public Const BCF_DOB = 10
Public Const BCF_SEX = 11
Public Const BCF_ISSUE = 12
```

Public Const BCF\_HEIGHT = 13  
 Public Const BCF\_WEIGHT = 14  
 Public Const BCF\_EYES = 15  
 Public Const BCF\_HAIR = 16  
 Public Const BCF\_SSNUMBER = 17  
 Public Const BCF\_PERMIT\_CLASS = 18  
 Public Const BCF\_PERMIT\_EXP = 19  
 Public Const BCF\_PERMIT\_ID = 20  
 Public Const BCF\_PERMIT\_ISSUE = 21  
 Public Const BCF\_PERMIT\_REST = 22  
 Public Const BCF\_PERMIT\_ENDORSEMENT = 23  
 Public Const BCF\_DRIVER\_LAST\_NAME = 24  
 Public Const BCF\_DRIVER\_FIRST\_NAME = 25  
 Public Const BCF\_DRIVER\_MIDDLE\_NAME = 26  
 Public Const BCF\_DRIVER\_NAME\_SUFFIX = 27  
 Public Const BCF\_DRIVER\_NAME\_PREFIX = 28  
 Public Const BCF\_MAIL\_STREET\_ADDRESS2 = 29  
 Public Const BCF\_RES\_STREET\_ADDRESS1 = 30  
 Public Const BCF\_RES\_STREET\_ADDRESS2 = 31  
 Public Const BCF\_RES\_CITY = 32  
 Public Const BCF\_RES\_JURISTRDICTION = 33  
 Public Const BCF\_RES\_POSTAL\_CODE = 34  
 Public Const BCF\_HEIGHT\_CM = 35  
 Public Const BCF\_WEIGHT\_KG = 36  
 Public Const BCF\_ISSUE\_TIMESTAMP = 37  
 Public Const BCF\_NUM\_OF\_DUPLICATES = 38  
 Public Const BCF\_MEDICAL\_IND = 39  
 Public Const BCF\_ORGAN\_DONOR = 40  
 Public Const BCF\_NON\_RESIDENT = 41  
 Public Const BCF\_UNIWUE\_CUSTOMER\_ID = 42  
 Public Const BCF\_AKA\_DOB = 43  
 Public Const BCF\_AKA\_SSN = 44  
 Public Const BCF\_AKA\_NAME = 45  
 Public Const BCF\_AKA\_LAST\_NAME = 46  
 Public Const BCF\_AKA\_FIRST\_NAME = 47  
 Public Const BCF\_AKA\_MIDDLE\_NAME = 48  
 Public Const BCF\_AKA\_SUFFIX = 49  
 Public Const BCF\_AKA\_PREFIX = 50

' emulated field indexes. Although PDF417 standard has fixed fields definition, the  
 ' fields population depends on the type of state. The following emulated fields  
 ' extract the data from the raw fields and organized it in unified manner regardless  
 ' to the state type for convenient integration  
 Public Const BCF\_EMUL\_FULL\_NAME = 100  
 Public Const BCF\_EMUL\_FIRST\_NAME = 101  
 Public Const BCF\_EMUL\_MIDDLE\_NAME = 102  
 Public Const BCF\_EMUL\_LAST\_NAME = 103  
 Public Const BCF\_EMUL\_NAME\_SUFFIX = 104

Public Const BCF\_EMUL\_DOB = 105  
 Public Const BCF\_EMUL\_ISSUE = 106  
 Public Const BCF\_EMUL\_EXP = 107  
 Public Const BCF\_EMUL\_ADDRESS = 108  
 Public Const BCF\_EMUL\_CITY = 109  
 Public Const BCF\_EMUL\_STATE = 110  
 Public Const BCF\_EMUL\_ZIP = 111  
 Public Const BCF\_EMUL\_LICENSE = 112  
 Public Const BCF\_EMUL\_SSN = 113

Public Const BCF\_EMUL\_END = 114  
 Public Const BCF\_EMUL\_EYES = 115  
 Public Const BCF\_EMUL\_HAIR = 116  
 Public Const BCF\_EMUL\_HEIGHT = 117  
 Public Const BCF\_EMUL\_WEIGHT = 118

' return error values  
 Public Const BC\_ERR\_NO\_BC\_FOUND = 0  
 Public Const BC\_ERR\_NONE = 1  
 Public Const BC\_ERR\_BAD\_PARAM = -1

## Appendix B – Supported States for Detection

Country Name	Region ID	Country Name	Country ID	Document Name	Document ID
USA	0	USA	0	Alabama	0
				Alaska	1
				Arizona	2
				Arkansas	3
				California	4
				Colorado	5
				Connecticut	6
				Delaware	7
				Washington D.C.	8
				Florida	9
				Georgia	10
				Idaho	11
				Illinois	12
				Indiana	13
				Iowa	14
				Kansas	15
Kentucky	16				

			Louisiana	17
			Maine	18
			Maryland	19
			Massachusetts	20
			Michigan	21
			Minnesota	22
			Mississippi	23
			Missouri	24
			Montana	25
			Nebraska	26
			Nevada	27
			New Hampshire	28
			New Jersey	29
			New Mexico	30
			New York	31
			North Carolina	32
			North Dakota	33
			Ohio	34
			Oklahoma	35
			Oregon	36
			Pennsylvania	37
			Rhode Island	38
			South Carolina	39
			South Dakota	40
			Tennessee	41
			Texas	42
			Utah	43
			Vermont	44
			Virginia	45
			Washington	46
			West Virginia	47
			Wisconsin	48
			Wyoming	49
			Hawaii	54
			Permanent Resident (Green Card)	81
			USA Army	82
			Social Security Card	83
			NY Police department	84

				Matricula consular (Mexican Id)	85
				Tribal	88
				US Virgin Islands	91
Australia	4	Australia	1	New South Wales - DL	50
				Australian Capital Territory - DL	51
				Queensland - DL	52
				Victoria - DL	53
				Tasmania - DL	55
				Western Australia - DL	56
				South Australia - DL	57
				Northern Territory - DL	58
				Cook Island – DL	59
				Fiji - DL	61
Asia	5	China	43	Hong Kong – National ID	470
		Indonesia	37	Indonesia – DL	410
		Malaysia	2	Malaysia – National ID + Driver License	60
		Singapore	14	Singapore – DL	180
		New Zealand	16	New Zealand	200
		United Arab Emirates	55	United Arab Emirates	610
		Brunei	68	Brunei	720
		Oman	85	Oman	900
		Qatar	86	Qatar	910
		Saudi Arabia	87	Saudi Arabia	930
		India	81	India	850
		Philippines	102	Philippines	1080
		Thailand	104	Thailand	1100
Canada	1	Canada	3	Ontario - DL	70
				Alberta - DL	71
				British Columbia - DL	72
				Manitoba - DL	73

				New Brunswick - DL	74
				New Foundland - DL	75
				New Territories - DL	76
				Nova Scotia - DL	77
				Saskatchewan - DL	78
				Canadian Citizen ID	79
				Prince Edward	880
				Quebec - DL	1079
South America	2	Chile	4	Chile - DL + National ID	80
		Mexico	6	Mexico – DL + National ID + Voter card	100
		Brazil	8	Brazil – DL + National ID	130
		Bermuda	13	Bermuda – DL + National ID	17
		Costa Rica	28	Costa Rica – National ID	320
		El Salvador	34	El Salvador – DL + National ID	380
		Guatemala	33	Guatemala – DL + National ID	370
		Nicaragua	32	Nicaragua – National ID	360
		Panama	36	Panama – National ID	400
		Peru	29	Peru – National ID	330
		Puerto Rico	30	Puerto Rico – DL + Firearms License	340
		Bolivia	60	Passports	670
		Columbia	79	Identity card	830
		Ecuador	67	Identity card	710
		Honduras	69	Identity card	730
		Dominican Republic	73	Identity card	770
		Haiti	74	Identity card	780
		Venezuela	80	DL	840
		Argentina	93	Identity card	990

		St. Christ Nevis	94	Identity card	1000
		Turks and Caicos Islands	103	Identity card	1090
		Belize	105	Identity card	1120
Europe	3	France	5	France – National ID	90
		Belgium	38	Belgium – National ID	420
		Bosnia	48	Bosnia – DL + National ID	530
		Bulgaria	45	Bulgaria – National ID	490
		Czech	57	Czech – DL	520
		Croatia	41	Croatia – DL + National ID	450
		Germany	10	Germany – DL	140
		Germany	10	Germany – National ID	141
		Spain	11	Spain – DL + National ID	150
		Bosnia	48	Bosnia – DL + National ID	530
		Hungary	49	Hungary – DL + National ID	540
		Holland	17	Holland – DL + National ID	210
		Italy	23	Identity card & driver license	270
		Luxembourg	18	Luxembourg – National ID	220
		Norway	15	Norway – DL	190
		Poland	27	Poland – DL + National ID	310
		Portugal	31	Portugal – National ID	350
		Romania	12	Romania – DL + National ID	160
		Slovakia	50	Slovakia – DL + National ID	550
		Sweden	22	Sweden driver license	260
		Switzerland	20	Switzerland –	240

				National ID	
		Turkey	25	Turkey – DL	290
		Israel	9	Israel – DL	120
		United Kingdom and Ireland	7	United Kingdom and Ireland – DL	110
		Kosovo	51	Kosovo – National ID	560
		Hungary	49	Identity card & driver license	540
		Slovenia	53	Slovenia Identity card	580
		Austria	57	Passports	640
		Liechtenstein	63	Identity card	680
		Finland	64	Identity card	690
		Estonia	71	Identity card	750
		Denmark	72	Identity card	760
		Iceland	78	Identity card	820
		Cyprus	76	Identity card	800
		Andorra	88	Identity card	940
		Guernsey	89	Identity card	950
		Isle of man	90	Identity card	960
		Latvia	91	Identity card	970
		Malta	92	Identity card	980
		Albania	95	Identity card	1010
		Montenegro	97	Identity card	1030
		Serbia	59	Identity card	660
		Macedonia	100	Identity card	1060
		Moldova	105	Identity card	1110
Africa	7	Namibia	82	Namibia	860
		South Africa	35	South Africa	390
		Zambia	83	Zambia	870
		Kenya	98	Kenya	1040
		Nigeria	99	Nigeria	1050
		Morocco	101	Morocco	1070
General Documents	6	University documents (USA)	24	Student Id (UMASS, Boston Un., Emerson Clg., Harward Un., NorthEastern Un., Suffolk Un.)	280
		Employment card	26	Employment card	300
		Service card	39	Service card	430



	Entertainment Card	40	Entertainment Card	440
	USA Pilots Access Card	42	USA Pilots Access Card	460
	OCB Cards	44	OCB Cards	480
	Spain Police	52	Spain Police	570
	EHIC	61	EHIC	630
	SCSIUSAC	65	SCSIUSAC	700
	USAA	70	USAA_CARDS	740
	PH Cards	75	PH Cards	790
	AMPORT Cards	84	AMPORT Cards	810
	Ireland Firearm	77	Ireland Firearm	810
		96		1020