Swelio Library

Belgian Electronic ID card access library

Table of Contents

Index a

1 Symbol Reference

1.1 Functions

The following table lists functions in this documentation.

Functions

	Name	Description
	ActivateCard (see page 9)	Established communication between the card and the reader
	ActivateCardEx (see page 10)	Established communication between the card and the reader
	AddFileToContainer (see page 10)	Add existing file to the container
	AddRemoveMessageFilter (see page 11)	Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.
	AllocateBuffer (see page 11)	Allocates the buffer in memory
	AllocateDefaultHWNDA (see page 12)	This function creates the invisible tool window
	AllocateDefaultHWNDW (see page 12)	This function creates the invisible tool window
	AllocateHWNDA (see page 12)	This function creates the invisible tool window using the provided window procedure
	AllocateHWNDW (see page 13)	This function creates the invisible tool window using the provided window procedure
	AllocateLayeredWindowA (see page 13)	This function creates the layered window using the provided window class name
	AllocateLayeredWindowW (see page 13)	This function creates the layered window using the provided window class name
	AllocateWindowClassA (see page 14)	This function creates the standard window using the provided window class name
	AllocateWindowClassW (see page 14)	This function creates the standard window using the provided window class name
	AlphaBlendBitmap (see page 14)	This is function AlphaBlendBitmap.
	AlphaBlendNative (see page 15)	This is function AlphaBlendNative.
	BringWindowToFront (see page 15)	This function brings the specified window to the top of the z-order.
	CardDecryptFileA (see page 15)	Decrypt file using Belgian Id card
	CardDecryptFileW (see page 15)	Decrypt file using Belgian Id card
	CardEncryptFileA (see page 16)	Encrypt file using Belgian Id card
	CardEncryptFileW (see page 16)	Encrypt file using Belgian Id card
	CardSignCadesT (see page 17)	Sign data with eID card according to CADES-T standard
	CardSignCMS (see page 17)	Sign data with eID card according to CMS standard
	CertSignCadesT (see page 18)	Sign data with the certificate file according to CADES-T standard
	CertSignCMS (see page 18)	Sign data with the certificate file according to CMS standard
	CheckMD5 (see page 19)	Checks the MD5 hash value of the memory buffer
	CheckSHA1 (see page 19)	Checks the SHA1 hash value of the memory buffer
	CheckSHA256 (see page 20)	Checks the SHA256 hash value of the memory buffer

	ClearFileAttributesA (see page 20)	This function sets the file attributes to normal.
	ClearFileAttributesW (see page 20)	This function sets the file attributes to normal.
	ClearUnusedMemory (see page 21)	Clears unused memory and minimized the application memory usage
	CloneFont (see page 21)	This is function CloneFont.
	ContainerCertificate (see page 21)	Assign certificate for signing ASIC container
	ContainerEidCertificate (see page 22)	Select EID card certificate to sign ASIC container
	ContainerPickCertificate (see page 22)	Pick certificate to sign ASIC container
	CopyNativeBitmap (see page 22)	This is function CopyNativeBitmap.
	CreateCardBuffer (see page 23)	Creates XML buffer
	CreateNativeBitmap (see page 23)	This is function CreateNativeBitmap.
	CreateUnicodeFileA (see page 23)	Creates UNICODE file
	CreateUnicodeFileW (see page 23)	Creates UNICODE file
	CreateWindowsFont (see page 24)	This is function CreateWindowsFont.
	CurrentIPAddressA (see page 24)	Returns the IP address
	CurrentIPAddressW (see page 24)	Returns the IP address
	DeactivateCard (see page 24)	Terminates a cennection between a smart card and a reader
	DeactivateCardEx (see page 25)	Terminates a connection between a smart card and a reader
	DeallocateBuffer (see page 25)	Deallocates the memory buffer
	DeallocateHWNDA (see page 25)	This function destroys the specified window.
	DeallocateHWNDW (see page 26)	This function destroys the specified window.
	DecryptFileAESA (see page 26)	Decrypts file using AES algorithm.
	DecryptFileAESW (see page 27)	Decrypts file using AES algorithm.
	DeleteCardBuffer (see page 27)	Deletes XML buffer
	DeleteToRecycleBinA (see page 27)	Deletes file to the Windows Recycle Bin
	DeleteToRecycleBinW (see page 28)	Deletes file to WIndows Recycle Bin
	DestroyFont (see page 28)	This is function DestroyFont.
	DestroyImageBuffer (see page 28)	Destroys the memory buffer
	DirectoryExistsA (see page 29)	Determines whether a specified directory exists.
	DirectoryExistsW (see page 29)	Determines whether a specified directory exists.
	DisplayCertificate (see page 30)	Displays the dialog window with certificate information
	DpiY (see page 30)	This is function DpiY.
	DrawAlphaText (see page 30)	This is function DrawAlphaText.
	DrawAlphaTextRect (see page 30)	This is function DrawAlphaTextRect.
	DrawLayeredWindow (see page 31)	Repaints the surface of the layered window
	DrawNativeBitmap (see page 31)	This is function DrawNativeBitmap.
	DrawTextDirect (see page 31)	This is function DrawTextDirect.

	DrawTextDirectEx (see page 32)	This is function DrawTextDirectEx.
	DrawTextGlow (see page 32)	This is function DrawTextGlow.
	DrawTextLine (see page 32)	This is function DrawTextLine.
	DrawTextOutline (see page 32)	This is function DrawTextOutline.
	DrawTextRect (see page 33)	This is function DrawTextRect.
	EmptyRecycleBin (see page 33)	Empties the recycle bin
	EmToPixels (see page 33)	This is function EmToPixels.
	EncodeCertificate (see page 33)	Performs Base64 encoding of the certificate
	EncodePhoto (see page 34)	Performs Base64 encoding of the photo
	EncryptFileAESA (see page 34)	Encrypts file using AES algorithm.
	EncryptFileAESW (see page 35)	Encrypts file using AES algorithm.
	FileCloseA (see page 35)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
	FileCloseW (see page 35)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
	FileCopyA (see page 36)	The CopyFile function copies an existing file to a new file.
	FileCopyW (see page 36)	The CopyFile function copies an existing file to a new file.
	FileCreateRewriteA (see page 37)	Creates new or overwrites existing file
	FileCreateRewriteW (see page 37)	Creates new or overwrites existing file
	FileDeleteA (see page 37)	Deletes a file from disk.
	FileDeleteW (see page 38)	Deletes a file from disk.
	FileExistsA (see page 38)	Tests whether a specified file exists.
	FileExistsW (see page 38)	Tests whether a specified file exists.
	FileExtensionIsA (see page 39)	Checks the file extension
	FileExtensionIsW (see page 39)	Checks the file extension
	FileGetSizeA (see page 39)	Retrieves the size of a specified file.
	FileGetSizeW (see page 40)	Retrieves the size of a specified file.
	FileIsExeA (see page 40)	Checks if the file is a Windows executable
	FileIsExeW (see page 41)	Checks if the file is a Windows executable
	FileIsIconA (see page 41)	Checks if the file is a Windows icon (.ico) file
	FileIsIconW (see page 41)	Checks if the file is a Windows icon (.ico) file
	FileIsImageA (see page 42)	Checks if the file is an image file
	FileIsImageW (see page 42)	Checks if the file is an image file
	FileIsLink (see page 42)	Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).
	FileOrFolderExistsA (see page 43)	Checks if the file or folder with the given name exists
	FileOrFolderExistsW (see page 43)	Checks if the file or folder with the given name exists
	FileRenameA (see page 43)	Renames the file
	FileRenameW (see page 44)	Renames the file
	FileWriteA (see page 44)	Writes string to the file
	FileWriteCharA (see page 44)	Writes one character to the file
	FileWriteCharW (see page 45)	Writes one character to the file
	FileWriteNewLineA (see page 45)	Writes new line sequence to the file
	FileWriteNewLineW (see page 45)	Writes new line sequence to the file
	FileWriteW (see page 46)	Writes string to the file
	fpreset (see page 46)	This is function fpreset.
	FreeContainer (see page 46)	Deallocates ASIC container

	FullPathA (see page 47)	Gets the full path to the file based on file name
	FullPathW (see page 47)	Gets the full path to the file based on file name
	GenerateAuthenticationSignatureA (see page 47)	Generate authentication signature
	GenerateAuthenticationSignatureExA (see page 48)	Generate authentication signature
	GenerateAuthenticationSignatureExW (see page 48)	Generate authentication signature
	GenerateAuthenticationSignatureW (see page 49)	Generate authentication signature
	GenerateBMPA (see page 49)	Generates Windows Bitmap file with QR Code image
	GenerateBMPW (see page 50)	Generates Windows Bitmap file with QR Code image
	GenerateNonRepudiationSignatureA (see page 50)	Generate non repudiation signature
	GenerateNonRepudiationSignatureExA (see page 51)	Generate non repudiation signature
	GenerateNonRepudiationSignatureExW (see page 51)	Generate non repudiation signature
	GenerateNonRepudiationSignatureW (see page 52)	Generate non repudiation signature
	GeneratePNGA (see page 52)	Generates PNG file with QR Code image
	GeneratePNGW (see page 53)	Generates PNG file with QR Code image
	GenerateQRCodeA (see page 53)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeExA (see page 53)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeExW (see page 54)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeW (see page 54)	Read eID card and save the identity information and address to PNG QR Code file
	GetAllFiles (see page 55)	Returns the names of files in a specified directory.
	GetCardBufferA (see page 55)	Gets XML or CSV information from the memory buffer
	GetCardBufferSize (see page 56)	This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format
	GetCardBufferW (see page 56)	Gets XML or CSV information from the memory buffer
	GetCardSerialNumber (see page 56)	Get the serial number of EID card
	GetCardVersion (see page 57)	Get the applet version number for card in the reader with specified number
	GetEncodedCertificateSize (see page 57)	Returns the size of the Base64 encoded certificate
	GetEncodedPhotoSize (see page 58)	Calculates buffer size for Base64 encoded photo
	GetFileMD5A (see page 58)	Gets the MD5 hash value for the file
	GetFileMD5W (see page 58)	Gets the MD5 hash value for the file
	GetFilesCountA (see page 59)	Calculates the number of files in the given folder
	GetFilesCountW (see page 59)	Calculates the number of files in the given folder
	GetFileSHA1A (see page 60)	Gets the SHA1 hash value for the file
	GetFileSHA1W (see page 60)	Gets the SHA1 hash value for the file
	GetFileSHA256A (see page 60)	Gets the SHA256 hash value for the file
	GetFileSHA256W (see page 61)	Gets the SHA256 hash value for the file
	GetHBitmapA (see page 61)	Generates Windows Bitmap in memory with QR Code image
	GetHBitmapW (see page 62)	Generates Windows Bitmap in memory with QR Code image
	GetISOCodeA (see page 62)	Returns the country ISO code based on the nationality string
	GetISOCodeW (see page 63)	Returns the country ISO code based on the nationality string

	GetMD5 (see page 63)	Gets the MD5 hash value for the content of the memory buffer
	GetPNGA (see page 64)	Writes PNG image to the memory buffer.
	GetPNGW (see page 64)	Writes PNG image to the memory buffer.
	GetReaderIndexA (see page 65)	Returns the zero-based reader index with specified name
	GetReaderIndexW (see page 65)	Returns the zero-based reader index with specified name
	GetReaderNameA (see page 65)	Returns the name of the card reader
	GetReaderNameLenA (see page 66)	Returns the length of the reader name
	GetReaderNameLenW (see page 66)	Returns the length of the reader name
	GetReaderNameW (see page 66)	Returns the name of the card reader
	GetReadersCount (see page 67)	Get number of card readers connected to PC
	GetSelectedReaderIndex (see page 67)	Returns the index of the active smart card reader
	GetSHA1 (see page 68)	Gets the SHA1 hash value for the content of the memory buffer
	GetSHA256 (see page 68)	Gets the SHA256 hash value for the content of the memory buffer
	GetStartupA (see page 68)	Checks if the application is registered to run when Windows starts
	GetStartupW (see page 69)	Checks if the application is registered to run when Windows starts
	GetSupportSIS (see page 69)	Checks if the SIS cards are supported by the engine
	GetTextLineSize (see page 69)	This is function GetTextLineSize.
	GetTextSize (see page 70)	This is function GetTextSize.
	GetTextSizeEx (see page 70)	This is function GetTextSizeEx.
	HibernateWindows (see page 70)	Hibernates Windows
	InitializeContainer (see page 70)	Initializes ASIC container
	IsAnimatedGIFA (see page 71)	Checks if the file is an animated GIF image file
	IsAnimatedGIFW (see page 71)	Checks if the file is an animated GIF image file
	IsCardActivated (see page 71)	Checks the connection between a smart card and a reader
	IsCardActivatedEx (see page 72)	Checks the connection between a smart card and a reader
	IsCardPresent (see page 72)	Checks if the card is present in the card reader
	IsCardPresentEx (see page 72)	Checks if the card is present in the card reader
	IsCardStillInserted (see page 73)	Checks if the card is still inserted in the card reader
	IsCardStillInsertedEx (see page 73)	Checks if the card is still inserted in the card reader
	IsCitrixSession (see page 73)	Checks if application is running in Citrix session
	IsConnectedToInternet (see page 74)	Checks if PC is connected to Internet
	IsDirectoryA (see page 74)	Verifies that a path is a valid directory.
	IsDirectoryW (see page 74)	Verifies that a path is a valid directory.
	IsEIDCard (see page 74)	Check if Belgian EID card is inserted into card reader
	IsEIDCardEx (see page 75)	Check if Belgian EID card is inserted into card reader
	IsEngineActive (see page 75)	Checks if the Swelio Engine is activated
	IsFemaleA (see page 76)	Checks if the card owner is female
	IsFemaleW (see page 76)	Checks if the card owner is female
	IsMaleA (see page 76)	Checks if the card owner is male
	IsMaleW (see page 77)	Checks if the card owner is male
	IsMediaCenter (see page 77)	Checks if the Media Center version of Windows is installed
	IsMetroActive (see page 77)	Checks if metro interface is active
	IsMultiTouchReady (see page 78)	Checks if the system is multi touch ready
	IsNativeWin64 (see page 78)	Checks if the application is native 64 bit executable
	IsRemoteSession (see page 78)	Checks if application is running in RDP session
	IsSISCard (see page 78)	Check if Belgian SIS card is inserted into card reader
	IsSISCardEx (see page 79)	Check if Belgian SIS card is inserted into card reader
	IsTabletPC (see page 79)	Checks if the application is running on the Tablet PC

	IsUnicodeFileA (see page 79)	Checks if the file is UNICODE file
	IsUnicodeFileW (see page 80)	Checks if the file is UNICODE file
	IsValidFileNameA (see page 80)	Checks if provided string is a valid file name
	IsValidFileNameW (see page 80)	Checks if provided string is a valid file name
	IsValidPathNameA (see page 81)	Checks if provided string is a valid file path
	IsValidPathNameW (see page 81)	Checks if provided string is a valid file path
	IsWindows10 (see page 82)	Checks if PC is running Windows 10 or better
	IsWindows7 (see page 82)	Checks if PC is running Windows 7 or better
	IsWindows8 (see page 82)	Checks if PC is Running Windows 8 or better
	IsWindowsVista (see page 82)	Checks if PC is running Windows Vista or better
	IsWindowsXP (see page 82)	Checks if PC is running Windows XP
	IsWindowsXPSP2 (see page 83)	Checks if PC is running Windows XP with Service Pack 2 installed
	IsWow64 (see page 83)	Checks if the 32 bit application runs on 64 bit Windows
	LayeredWndProcA (see page 83)	The default window procedure for the layered window
	LayeredWndProcW (see page 83)	The default window procedure for the layered window
	LoadBitmapJPG (see page 84)	This is function LoadBitmapJPG.
	LoadBitmapPNG (see page 84)	This is function LoadBitmapPNG.
	LoadCertificateA (see page 84)	Reads the certificate from a file
	LoadCertificateW (see page 84)	Reads the certificate from a file
	LoadIdentityA (see page 85)	Reads the raw identity information from a file
	LoadIdentityW (see page 85)	Reads the raw identity information from a file
	LoadPhotoA (see page 86)	Loads photo from a file
	LoadPhotoW (see page 86)	Loads photo from a file
	LoadPNGResource (see page 86)	This is function LoadPNGResource.
	MakeCompatibleBitmap (see page 87)	This is function MakeCompatibleBitmap.
	MakeSoundFromFileA (see page 87)	Plays the wave sound from the file
	MakeSoundFromFileW (see page 87)	Plays the wave sound from the file
	MakeSoundFromResourceA (see page 87)	Plays the wave sound from the resource
	MakeSoundFromResourceW (see page 88)	Plays the wave sound from the resource
	PointsToPixels (see page 88)	This is function PointsToPixels.
	PortAvailable (see page 88)	Checks if the port with specified number is available
	ReadAddressA (see page 89)	Read address information from Belgian eID card
	ReadAddressExA (see page 89)	Read address information from Belgian eID card
	ReadAddressExW (see page 89)	Read address information from Belgian eID card
	ReadAddressW (see page 90)	Read address information from Belgian eID card
	ReadAuthenticationCertificate (see page 90)	Read Authentication Certificate to memory
	ReadAuthenticationCertificateEx (see page 90)	Read Authentication Certificate to memory
	ReadBufferFromFileA (see page 91)	Reads the content of the file to the memory buffer
	ReadBufferFromFileW (see page 91)	Reads the content of the file to the memory buffer
	ReadCaCertificate (see page 92)	Read Ca Certificate to memory
	ReadCaCertificateEx (see page 92)	Read Ca Certificate to memory
	ReadIdentityA (see page 92)	Read identity information from Belgian eID card
	ReadIdentityExA (see page 93)	Read identity information from Belgian eID card
	ReadIdentityExW (see page 93)	Read identity information from Belgian eID card
	ReadIdentityW (see page 94)	Read identity information from Belgian eID card

	ReadNonRepudiationCertificate (see page 94)	Read Non Repudiation Certificate to memory
	ReadNonRepudiationCertificateEx (see page 94)	Read Non Repudiation Certificate to memory
	ReadPhoto (see page 95)	Reads a photo from a card
	ReadPhotoAsBitmap (see page 95)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
	ReadPhotoAsBitmapEx (see page 96)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the Windows bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
	ReadPhotoEx (see page 96)	Reads a photo from a card
	ReadRootCaCertificate (see page 96)	Read Root Ca Certificate to memory
	ReadRootCaCertificateEx (see page 97)	Read Root Ca Certificate to memory
	ReadRrnCertificate (see page 97)	Read Rrn Certificate to memory
	ReadRrnCertificateEx (see page 98)	Read Rrn Certificate to memory
	ReadSISCardA (see page 98)	Read Belgian SIS card.
	ReadSISCardExA (see page 98)	Read Belgian SIS card.
	ReadSISCardExW (see page 99)	Read Belgian SIS card.
	ReadSISCardW (see page 99)	Read Belgian SIS card.
	RecycleBinEmpty (see page 100)	Returns TRUE if Windows Recyce Bin is empty
	ReloadReadersList (see page 100)	Reloads the list of the available card readers
	RemoveCallback (see page 100)	Remove callback procedure for card events
	RemoveStartupA (see page 100)	Removes the application from the list of the automatically started applications
	RemoveStartupW (see page 101)	Removes the application from the list of the automatically started applications
	RestoreWindowSubclassA (see page 101)	Restores window standard procedure
	RestoreWindowSubclassW (see page 101)	Restores window standard procedure
	SaveAuthenticationCertificateA (see page 102)	Save Authentication Certificate to a file
	SaveAuthenticationCertificateExW (see page 102)	Save Authentication Certificate to a file
	SaveAuthenticationCertificateW (see page 102)	Save Authentication Certificate to a file
	SaveCaCertificateA (see page 103)	Save Ca Certificate to a file
	SaveCaCertificateExW (see page 103)	Save Ca Certificate to a file
	SaveCaCertificateW (see page 103)	Save Ca Certificate to a file
	SaveCardToToXMLStreamExA (see page 104)	Read eID card and save the information to XML buffer
	SaveCardToToXMLStreamExW (see page 104)	Read eID card and save the information to XML buffer
	SaveCardToXmlA (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlExA (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlExW (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlW (see page 106)	Read eID card and save the information to XML file
	SaveContainer (see page 106)	Save container to the file
	SaveIdentityA (see page 107)	Saves indentity infornation to a file

	SaveIdentityW (see page 107)	Saves indentity infornation to a file
	SaveNonRepudiationCertificateA (see page 107)	Save Non Repudiation Certificate to a file
	SaveNonRepudiationCertificateExW (see page 108)	Save Non Repudiation Certificate to a file
	SaveNonRepudiationCertificateW (see page 108)	Save Non Repudiation Certificate to a file
	SavePersonCsvToStreamA (see page 108)	Read eID card and save the identity information to CSV memory buffer
	SavePersonCsvToStreamW (see page 109)	Read eID card and save the identity information to CSV memory buffer
	SavePersonToCsvA (see page 109)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvExA (see page 110)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvExW (see page 110)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvW (see page 111)	Read eID card and save the identity information and address to CSV file
	SavePhotoA (see page 111)	Save photo to a file
	SavePhotoAsBitmapA (see page 111)	Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsBitmapExA (see page 112)	Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsBitmapExW (see page 112)	Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsBitmapW (see page 113)	Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegA (see page 113)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegExA (see page 113)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegExW (see page 114)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegW (see page 114)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoW (see page 114)	Saves photo to a file
	SaveRootCaCertificateA (see page 115)	Save Root Ca Certificate to a file
	SaveRootCaCertificateExW (see page 115)	Save Root Ca Certificate to a file

	SaveRootCaCertificateW (see page 116)	Save Root Ca Certificate to a file
	SaveRrnCertificateA (see page 116)	Save RRN Certificate to a file
	SaveRrnCertificateExW (see page 116)	Save RRN Certificate to a file
	SaveRrnCertificateW (see page 117)	Save RRN Certificate to a file
	SelectReader (see page 117)	When more than 1 reader connected, select the reader with specified number
	SelectReaderByNameA (see page 117)	Select active smart card reader by providing the reader name
	SelectReaderByNameW (see page 118)	Select active smart card reader by providing the reader name
	SendAPDU (see page 118)	This is function SendAPDU.
	SetCallback (see page 118)	Activates callback procedure for card status change event
	SetMWCompatibility (see page 119)	Set the compatibility mode with the old version of the oficial EID MiddleWare
	SetStartupA (see page 119)	Register application to run when Windows starts
	SetStartupW (see page 119)	Register application to run when Windows starts
	SetSupportSIS (see page 120)	Activates or deactivates SIS card support by engine
	ShellCopyFileA (see page 120)	Copies file to the new location
	ShellCopyFileW (see page 120)	Copies file to the new location
	ShutdownWindows (see page 121)	Logs off the interactive user, shuts down the system.
	StartEngine (see page 121)	Activates the Swelio Engine.
	StopEngine (see page 122)	Deactivates the Swelio Engine
	StretchNativeBitmap (see page 122)	This is function StretchNativeBitmap.
	StripFileNameA (see page 122)	Replaces environment variable names with values
	StripFileNameW (see page 123)	Replaces environment variable names with values
	SuspendWindows (see page 123)	Suspends Windows
	TurnMonitorOff (see page 123)	Turns the monitor off
	TurnMonitorOn (see page 123)	Turns the monitor on
	UpdateWindowPosition (see page 124)	Updated the window position
	VerifyPinA (see page 124)	Verify PIN code
	VerifyPinExA (see page 124)	Verify PIN code
	VerifyPinExW (see page 125)	Verify PIN code
	VerifyPinW (see page 125)	Verify PIN code
	VerifySignature (see page 125)	Verifies the signature from the specified hash value.
	WriteBufferToFileA (see page 126)	Writes the memory buffer to file
	WriteBufferToFileW (see page 126)	Writes the memory buffer to file

1.1.1 ActivateCard Function

Established communication between the card and the reader

C++

```
BOOL WINAPI ActivateCard();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE if the card is activated, otherwise returns FALSE

Description

The ActivateCard function establishes a connection between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

Example

```
if (IsCardPresent())
{
   ActivateCard();
}
```

1.1.2 ActivateCardEx Function

Established communication between the card and the reader

C++

```
BOOL WINAPI ActivateCardEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Returns

Returns TRUE if the card is activated, otherwise returns FALSE

Description

The ActivateCard (see page 9) function establishes a connection between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned.

1.1.3 AddFileToContainer Function

Add existing file to the container

C++

```
BOOL WINAPI AddFileToContainer(LPVOID container, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function
LPSTR fileName	The name of the file which will be added to the container

Returns

Returns true if the operation is successful, otherwise returns false

Description

Call this function to include the file to the container. The file must exist.

1.1.4 AddRemoveMessageFilter Function

Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.

C++

```
void WINAPI AddRemoveMessageFilter(UINT message, DWORD dwFlags);
```

File

System.h (see page 170)

Parameters

Parameters	Description
UINT message	Specifies the message to add to or remove from the filter.
DWORD dwFlags	Specifies the action to be performed. One of the following values. • MSGFLT_ADD - Adds the message to the filter. This has the effect of allowing the message to be received. • MSGFLT_REMOVE - Removes the message from the filter. This has the effect of blocking the message.

Description

This function changes the message filter for Windows Vista or better. UIPI is a security feature that prevents messages from being received from a lower integrity level sender. All such messages with a value above WM_USER are blocked by default. The filter, somewhat contrary to intuition, is a list of messages that are allowed through. Therefore, adding a message to the filter allows that message to be received from a lower integrity sender, while removing a message blocks that message from being received.

Certain messages with a value less than WM_USER are required to pass through the filter regardless of the filter setting. You can call this function to remove one of those messages from the filter and it will return TRUE. However, the message will still be received by the calling process.

1.1.5 AllocateBuffer Function

Allocates the buffer in memory

C++

```
void* WINAPI AllocateBuffer(int bufferSize);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
int bufferSize	The size of the buffer

Returns

The pointer to the allocated memory block

Description

AllocateBuffer allocates a block of the given size on the heap, and returns the address of this memory.The bytes of the allocated buffer are not set to zero. To dispose of the buffer, use DeallocateBuffer (see page 25) function.

1.1.6 AllocateDefaultHWNDA Function

This function creates the invisible tool window

C++

```
HWND WINAPI AllocateDefaultHWNDA();
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.7 AllocateDefaultHWNDW Function

This function creates the invisible tool window

C++

```
HWND WINAPI AllocateDefaultHWNDW();
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.8 AllocateHWNDA Function

This function creates the invisible tool window using the provided window procedure

C++

```
HWND WINAPI AllocateHWNDA(LONG_PTR method);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.9 AllocateHWNDW Function

This function creates the invisible tool window using the provided window procedure

C++

```
HWND WINAPI AllocateHWNDW(LONG_PTR method);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the invisible zero-size tool window that can be used for internal purposes, like processing the special Windows messages for synchronization, etc...

1.1.10 AllocateLayeredWindowA Function

This function creates the layered window using the provided window class name

C++

```
HWND WINAPI AllocateLayeredWindowA(LPCSTR className);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the layered window using the provided window class name

1.1.11 AllocateLayeredWindowW Function

This function creates the layered window using the provided window class name

C++

```
HWND WINAPI AllocateLayeredWindowW(LPCWSTR className);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the layered window using the provided window class name

1.1.12 AllocateWindowClassA Function

This function creates the standard window using the provided window class name

C++

```
HWND WINAPI AllocateWindowClassA(LPCSTR className);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the standard window using the provided window class name

1.1.13 AllocateWindowClassW Function

This function creates the standard window using the provided window class name

C++

```
HWND WINAPI AllocateWindowClassW(LPCWSTR className);
```

File

System.h (see page 170)

Returns

If the function succeeds, the return value is a handle to the new window. If the function fails, the return value is NULL.

Description

This function creates the standard window using the provided window class name

1.1.14 AlphaBlendBitmap Function

This is function AlphaBlendBitmap.

C++

```
void WINAPI AlphaBlendBitmap(HBITMAP src, HDC hdc, int left, int top, int width, int
height, int alpha);
```

File

Graphics.h (see page 164)

1.1.15 AlphaBlendNative Function

This is function AlphaBlendNative.

C++

```
void WINAPI AlphaBlendNative(HDC hdcDest, int xoriginDest, int yoriginDest, int wDest, int hDest, HDC hdcSrc, int xoriginSrc, int yoriginSrc, int wSrc, int hSrc, BYTE alpha);
```

File

Graphics.h (see page 164)

1.1.16 BringWindowToFront Function

This function brings the specified window to the top of the z-order.

C++

```
void WINAPI BringWindowToFront(HWND window);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND window	Handle to the window to bring to the top of the z-order.

1.1.17 CardDecryptFileA Function

Decrypt file using Belgian Id card

C++

```
BOOL WINAPI CardDecryptFileA(LPSTR szSource, LPSTR szDestination);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR szSource	The name of the encrypted file
LPSTR szDestination	The name of the decrypted file

Description

Decrypt file which was encrypted using CardEncryptFile function

1.1.18 CardDecryptFileW Function

Decrypt file using Belgian Id card

C++

```
BOOL WINAPI CardDecryptFileW(LPWSTR szSource, LPWSTR szDestination);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR szSource	The name of the encrypted file
LPWSTR szDestination	The name of the decrypted file

Description

Decrypt file which was encrypted using CardEncryptFile function

1.1.19 CardEncryptFileA Function

Encrypt file using Belgian Id card

C++

```
BOOL WINAPI CardEncryptFileA(LPSTR szSource, LPSTR szDestination);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR szSource	The name of the source file
LPSTR szDestination	The name of the encrypted file

Description

Encrypt file using Belgian Id card. The card must be inserted in the reader

1.1.20 CardEncryptFileW Function

Encrypt file using Belgian Id card

C++

```
BOOL WINAPI CardEncryptFileW(LPWSTR szSource, LPWSTR szDestination);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR szSource	The name of the source file
LPWSTR szDestination	The name of the encrypted file

Description

Encrypt file using Belgian Id card. The card must be inserted in the reader

1.1.21 CardSignCadesT Function

Sign data with eID card according to CADES-T standard

C++

```
BOOL WINAPI CardSignCadesT(int readerNumber, BYTE * data, UINT dataLen, BYTE * signature,
UINT * signatureLen);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CADES-T signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.22 CardSignCMS Function

Sign data with eID card according to CMS standard

C++

```
BOOL WINAPI CardSignCMS(int readerNumber, BYTE * data, UINT dataLen, BYTE * signature, UINT
* signatureLen);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CMS signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.23 CertSignCadesT Function

Sign data with the certificate file according to CADES-T standard

C++

```
BOOL WINAPI CertSignCadesT(LPWSTR certificate, LPWSTR password, BYTE * data, UINT dataLen,
BYTE * signature, UINT * signatureLen);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR certificate	The name of the certificate file
LPWSTR password	The private key password
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CADES-T signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.24 CertSignCMS Function

Sign data with the certificate file according to CMS standard

C++

```
BOOL WINAPI CertSignCMS(LPWSTR certificate, LPWSTR password, BYTE * data, UINT dataLen,
BYTE * signature, UINT * signatureLen);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR certificate	The name of the certificate file
LPWSTR password	The private key password
BYTE * data	the data to sign
UINT dataLen	the size of the data buffer
BYTE * signature	the signature buffer
UINT * signatureLen	the size of the signature buffer

Returns

Returns true if the operation is successful, otherwise returns false

Description

Create CMS signature for data buffer. Can be used for digital signature of PDF documents in combination with external PDF library

1.1.25 CheckMD5 Function

Checks the MD5 hash value of the memory buffer

C++

```
BOOL WINAPI CheckMD5(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer
BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.26 CheckSHA1 Function

Checks the SHA1 hash value of the memory buffer

C++

```
BOOL WINAPI CheckSHA1(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer
BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.27 CheckSHA256 Function

Checks the SHA256 hash value of the memory buffer

C++

```
BOOL WINAPI CheckSHA256(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source bytes
int sourceSize	The size of the source buffer
BYTE* buffer	The hash value buffer
int bufferSize	The size of the hash value buffer

Returns

Returns TRUE if the hash value is correct, otherwise returns false

Description

This function checks if the provided value of the hash is valid

1.1.28 ClearFileAttributesA Function

This function sets the file attributes to normal.

C++

```
void WINAPI ClearFileAttributesA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Description

This function removed additional file attributes, like system, read-only and hidden.

1.1.29 ClearFileAttributesW Function

This function sets the file attributes to normal.

C++

```
void WINAPI ClearFileAttributesW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Description

This function removed additional file attributes, like system, read-only and hidden.

1.1.30 ClearUnusedMemory Function

Clears unused memory and minimized the application memory usage

C++

```
void WINAPI ClearUnusedMemory();
```

File

System.h (see page 170)

1.1.31 CloneFont Function

This is function CloneFont.

C++

```
HFONT WINAPI CloneFont(HFONT hFont);
```

File

Graphics.h (see page 164)

1.1.32 ContainerCertificate Function

Assign certificate for signing ASIC container

C++

```
BOOL WINAPI ContainerCertificate(LPVOID container, LPWSTR fileName, LPWSTR password);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function

Returns

Returns true if the operation is successful, otherwise returns false

Description

Use this function to assign certificate which will be used to sign container at the moment when container will be saved to the file

1.1.33 ContainerEidCertificate Function

Select EID card certificate to sign ASIC container

C++

```
BOOL WINAPI ContainerEidCertificate(LPVOID container, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function

Returns

Returns true if the operation is successful, otherwise returns false

Description

Call this function to select EID certificate to sign ASIC container

1.1.34 ContainerPickCertificate Function

Pick certificate to sign ASIC container

C++

```
BOOL WINAPI ContainerPickCertificate(LPVOID container);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function

Returns

Returns true if the operation is successful, otherwise returns false

Description

Use this function to let user pick the certificate to sign ASIC contaner The dialog to choose the certificate will be shown to the user

1.1.35 CopyNativeBitmap Function

This is function CopyNativeBitmap.

C++

```
void WINAPI CopyNativeBitmap(HBITMAP src, HDC dstDC, int width, int height, int left, int top);
```

File

Graphics.h (see page 164)

1.1.36 CreateCardBuffer Function

Creates XML buffer

C++

```
void* WINAPI CreateCardBuffer();
```

File

Swelio.h (see page 165)

Returns

The memory buffer to store information

Description

Use this function to create XML buffer

1.1.37 CreateNativeBitmap Function

This is function CreateNativeBitmap.

C++

```
HBITMAP WINAPI CreateNativeBitmap(INT width, INT height, __deref_opt_out void ** ppvBits);
```

File

Graphics.h (see page 164)

1.1.38 CreateUnicodeFileA Function

Creates UNICODE file

C++

```
void WINAPI CreateUnicodeFileA(LPCSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

1.1.39 CreateUnicodeFileW Function

Creates UNICODE file

C++

```
void WINAPI CreateUnicodeFileW(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

1.1.40 CreateWindowsFont Function

This is function CreateWindowsFont.

C++

```
HFONT WINAPI CreateWindowsFont(LPCWSTR fontFamily, INT size, INT fontStyle, INT
fontQuality);
```

File

Graphics.h (see page 164)

1.1.41 CurrentIPAddressA Function

Returns the IP address

C++

```
void WINAPI CurrentIPAddressA(LPSTR address, UINT len);
```

File

SystemInfo.h (see page 171)

1.1.42 CurrentIPAddressW Function

Returns the IP address

C++

```
void WINAPI CurrentIPAddressW(LPWSTR address, UINT len);
```

File

SystemInfo.h (see page 171)

1.1.43 DeactivateCard Function

Terminates a cennection between a smart card and a reader

C++

```
void WINAPI DeactivateCard();
```

File

Swelio.h (see page 165)

Description

The DeactivateCard function terminates a connection previously opened between the calling application and a smart card in the target reader.

1.1.44 DeactivateCardEx Function

Terminates a connection between a smart card and a reader

C++

```
void WINAPI DeactivateCardEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

The DeactivateCard (see page 24) function terminates a connection previously opened between the calling application and a smart card in the target reader.

1.1.45 DeallocateBuffer Function

Deallocates the memory buffer

C++

```
void WINAPI DeallocateBuffer(void* buffer);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
void* buffer	The pointer to the memory buffer

Description

DeallocateBuffer frees a memory block previously allocated with AllocateBuffer (see page 11). Use this procedure to dispose of a memory block obtained with AllocateBuffer (see page 11).

1.1.46 DeallocateHWNDA Function

This function destroys the specified window.

C++

```
BOOL WINAPI DeallocateHWNDA(HWND hwnd);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND hwnd	Handle to the window to be destroyed

Description

This function restores the window default procedure and destroys the window

1.1.47 DeallocateHWNDW Function

This function destroys the specified window.

C++

```
BOOL WINAPI DeallocateHWNDW(HWND hwnd);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND hwnd	Handle to the window to be destroyed

Description

This function restores the window default procedure and destroys the window

1.1.48 DecryptFileAESA Function

Decrypts file using AES algorithm.

C++

```
BOOL WINAPI DecryptFileAESA(LPSTR szSource, LPSTR szDestination, LPSTR szPassword);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR szSource	The source file name
LPSTR szDestination	The decrypted file name
LPSTR szPassword	The password

Returns

Returns TRUE if the file is successfully decrypted, otherwise returns FALSE.

Description

Use this function to decrypt the file using AES algorithm

1.1.49 DecryptFileAESW Function

Decrypts file using AES algorithm.

C++

```
BOOL WINAPI DecryptFileAESW(LPWSTR szSource, LPWSTR szDestination, LPWSTR szPassword);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR szSource	The source file name
LPWSTR szDestination	The decrypted file name
LPWSTR szPassword	The password

Returns

Returns TRUE if the file is successfully decrypted, otherwise returns FALSE.

Description

Use this function to decrypt the file using AES algorithm

1.1.50 DeleteCardBuffer Function

Deletes XML buffer

C++

```
void WINAPI DeleteCardBuffer(void* buffer);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
void* buffer	The memory buffer to store information

Description

Use this function to delete the buffer

1.1.51 DeleteToRecycleBinA Function

Deletes file to the Windows Recycle Bin

C++

```
void WINAPI DeleteToRecycleBinA(LPSTR fileName, BOOL silent);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BOOL silent	Do not display a progress dialog box

Description

Use this function to delete the file to Windows Recycle Bin

1.1.52 DeleteToRecycleBinW Function

Deletes file to WIndows Recycle Bin

C++

```
void WINAPI DeleteToRecycleBinW(LPWSTR fileName, BOOL silent);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BOOL silent	Do not display a progress dialog box

Description

Use this function to delete the file to Windows Recycle Bin

1.1.53 DestroyFont Function

This is function DestroyFont.

C++

```
void WINAPI DestroyFont(HFONT hFont);
```

File

Graphics.h (see page 164)

1.1.54 DestroyImageBuffer Function

Destroys the memory buffer

C++

```
void WINAPI DestroyImageBuffer(void* buffer);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
void* buffer	The memory buffer

Description

Destroys the memory buffer created to hold the image returned by GetPNGA (see page 64) (Ansi) or GetPNGW (see page 64) (Unicode) functions

1.1.55 DirectoryExistsA Function

Determines whether a specified directory exists.

C++

```
BOOL WINAPI DirectoryExistsA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the directory

Returns

Returns TRUE if the directory exists, otherwise returns FALSE

Description

Call DirectoryExists to determine whether the directory specified by the Directory parameter exists. If the directory exists, the function returns True. If the directory does not exist, the function returns False. If a full path name is entered, DirectoryExists searches for the directory along the designated path. Otherwise, the Directory parameter is interpreted as a relative path name from the current directory.

1.1.56 DirectoryExistsW Function

Determines whether a specified directory exists.

C++

```
BOOL WINAPI DirectoryExistsW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the directory

Returns

Returns TRUE if the directory exists, otherwise returns FALSE

Description

Call DirectoryExists to determine whether the directory specified by the Directory parameter exists. If the directory exists, the function returns True. If the directory does not exist, the function returns False. If a full path name is entered, DirectoryExists searches for the directory along the designated path. Otherwise, the Directory parameter is interpreted as a relative path name from the current directory.

1.1.57 DisplayCertificate Function

Displays the dialog window with certificate information

C++

```
void WINAPI DisplayCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidCertificate certificate	The certificate data

Description

Use this function to show the certificate for the user

1.1.58 DpiY Function

This is function DpiY.

C++

```
int WINAPI DpiY();
```

File

Graphics.h (see page 164)

1.1.59 DrawAlphaText Function

This is function DrawAlphaText.

C++

```
void WINAPI DrawAlphaText(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int width, int
height, UINT flags);
```

File

Graphics.h (see page 164)

1.1.60 DrawAlphaTextRect Function

This is function DrawAlphaTextRect.

C++

```
void WINAPI DrawAlphaTextRect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, LPRECT
lpRect, UINT flags);
```

File

Graphics.h (see page 164)

1.1.61 DrawLayeredWindow Function

Repaints the surface of the layered window

C++

```
void WINAPI DrawLayeredWindow(HWND handle, int left, int top, int width, int height, HDC buffer, COLORREF colorKey, byte alpha, BOOL redrawOnly);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND handle	The handle of the window
int left	The horizontal coordinate of the window
int top	The vertical coordinate of the window
int width	The width of the window
int height	The height of the window
HDC buffer	Handle to a DC for the surface that defines the layered window
COLORREF colorKey	COLORREF structure that specifies the color key to be used when composing the layered window.
byte alpha	Specifies an alpha transparency value to be used on the entire source bitmap
BOOL redrawOnly	Only redraw and do not update the window position

1.1.62 DrawNativeBitmap Function

This is function DrawNativeBitmap.

C++

```
void WINAPI DrawNativeBitmap(HBITMAP src, HBITMAP dst, int width, int height);
```

File

Graphics.h (see page 164)

1.1.63 DrawTextDirect Function

This is function DrawTextDirect.

C++

```
void WINAPI DrawTextDirect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int left, int top);
```

File

Graphics.h (see page 164)

1.1.64 DrawTextDirectEx Function

This is function DrawTextDirectEx.

C++

```
void WINAPI DrawTextDirectEx(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, COLORREF background, int left, int top);
```

File

Graphics.h (see page 164)

1.1.65 DrawTextGlow Function

This is function DrawTextGlow.

C++

```
void WINAPI DrawTextGlow(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF foreColor, int left, int top);
```

File

Graphics.h (see page 164)

1.1.66 DrawTextLine Function

This is function DrawTextLine.

C++

```
void WINAPI DrawTextLine(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, int left, int top);
```

File

Graphics.h (see page 164)

1.1.67 DrawTextOutline Function

This is function DrawTextOutline.

C++

```
void WINAPI DrawTextOutline(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF foreColor, COLORREF backColor, int left, int top);
```

File

Graphics.h (see page 164)

1.1.68 DrawTextRect Function

This is function DrawTextRect.

C++

```
void WINAPI DrawTextRect(HDC hDC, LPCWSTR s, HFONT hFont, COLORREF color, COLORREF
background, LPRECT lpRect, UINT flags);
```

File

Graphics.h (see page 164)

1.1.69 EmptyRecycleBin Function

Empties the recycle bin

C++

```
void WINAPI EmptyRecycleBin();
```

File

System.h (see page 170)

Description

Removes all files from the Windows recycle bin

1.1.70 EmToPixels Function

This is function EmToPixels.

C++

```
int WINAPI EmToPixels(int em);
```

File

Graphics.h (see page 164)

1.1.71 EncodeCertificate Function

Performs Base64 encoding of the certificate

C++

```
int WINAPI EncodeCertificate(PEidCertificate certificate, BYTE* buffer, int bufferSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidCertificate certificate	The certificate data
BYTE* buffer	The Base64 encoded certificate buffer

int bufferSize	The size of the buffer

Returns

Returns the size of the buffer needed to hold the encoded certificate

1.1.72 EncodePhoto Function

Performs Base64 encoding of the photo

C++

```
int WINAPI EncodePhoto(PeidPicture photo, BYTE* buffer, int bufferSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure
BYTE* buffer	The Base64 encoded photo buffer
int bufferSize	The size of the buffer

Returns

Returns the size of the buffer needed to hold the encoded photo

Description

Use this function for Base64 encoding of the photo

1.1.73 EncryptFileAESA Function

Encrypts file using AES algorithm.

C++

```
BOOL WINAPI EncryptFileAESA(LPSTR szSource, LPSTR szDestination, LPSTR szPassword);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR szSource	The source file name
LPSTR szDestination	The encrypted file name
LPSTR szPassword	The password

Returns

Returns TRUE if the file is successfully encrypted, otherwise returns FALSE

Description

Use this function to encrypt the file using AES algorithm

1.1.74 EncryptFileAESW Function

Encrypts file using AES algorithm.

C++

```
BOOL WINAPI EncryptFileAESW(LPWSTR szSource, LPWSTR szDestination, LPWSTR szPassword);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR szSource	The source file name
LPWSTR szDestination	The encrypted file name
LPWSTR szPassword	The password

Returns

Returns TRUE if the file is successfully encrypted, otherwise returns FALSE

Description

Use this function to encrypt the file using AES algorithm

1.1.75 FileCloseA Function

Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.

C++

```
void WINAPI FileCloseA(HANDLE handle);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

Description

Closes the file handle of the specified file when its not in use anymore

1.1.76 FileCloseW Function

Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.

C++

```
void WINAPI FileCloseW(HANDLE handle);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

Description

Closes the file handle of the specified file when its not in use anymore

1.1.77 FileCopyA Function

The CopyFile function copies an existing file to a new file.

C++

```
BOOL WINAPI FileCopyA(LPSTR oldName, LPSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR oldName	The name of the source file
LPSTR newName	The name of the destination file

Returns

The result of the function is TRUE when the file is successfully copied to the new location, otherwise the result is FALSE.

Description

This function makes a copy of the file with the new name or path.

1.1.78 FileCopyW Function

The CopyFile function copies an existing file to a new file.

C++

```
BOOL WINAPI FileCopyW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR oldName	The name of the source file
LPWSTR newName	The name of the destination file

Returns

The result of the function is TRUE when the file is successfully copied to the new location, otherwise the result is FALSE.

Description

This function makes a copy of the file with the new name or path.

1.1.79 FileCreateRewriteA Function

Creates new or overwrites existing file

C++

```
HANDLE WINAPI FileCreateRewriteA(LPCSTR fileName);
```

File

FileOperations.h (see page 162)

Returns

The result of the function is the handle of the file

Description

This function creates the new file with provided file name if the file with given name does not exists. If the file exists, it will be overwritten and the current content of the file will be lost

1.1.80 FileCreateRewriteW Function

Creates new or overwrites existing file

C++

```
HANDLE WINAPI FileCreateRewriteW(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Returns

The result of the function is the handle of the file

Description

This function creates the new file with provided file name if the file with given name does not exists. If the file exists, it will be overwritten and the current content of the file will be lost

1.1.81 FileDeleteA Function

Deletes a file from disk.

C++

```
void WINAPI FileDeleteA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Description

DeleteFile deletes the file named by fileName from the disk.

1.1.82 FileDeleteW Function

Deletes a file from disk.

C++

```
void WINAPI FileDeleteW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Description

DeleteFile deletes the file named by fileName from the disk.

1.1.83 FileExistsA Function

Tests whether a specified file exists.

C++

```
BOOL WINAPI FileExistsA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Returns

FileExists returns TRUE if the file specified by FileName exists. If the file does not exist, FileExists returns FALSE.

Description

Use this function to check if the file with provided name exists.

1.1.84 FileExistsW Function

Tests whether a specified file exists.

C++

```
BOOL WINAPI FileExistsW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Returns

FileExists returns TRUE if the file specified by FileName exists. If the file does not exist, FileExists returns FALSE.

Description

Use this function to check if the file with provided name exists.

1.1.85 FileExtensionIsA Function

Checks the file extension

C++

```
BOOL WINAPI FileExtensionIsA(LPCSTR fileName, LPCSTR ext);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file
LPCSTR ext	The file name extension

Returns

Returns true if the file has a specified extension, otherwise returns false.

Description

This function checks if the file has a given extension

1.1.86 FileExtensionIsW Function

Checks the file extension

C++

```
BOOL WINAPI FileExtensionIsW(LPCWSTR fileName, LPCWSTR ext);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file
LPCWSTR ext	The file name extension

Returns

Returns true if the file has a specified extension, otherwise returns false.

Description

This function checks if the file has a given extension

1.1.87 FileGetSizeA Function

Retrieves the size of a specified file.

C++

```
DWORD WINAPI FileGetSizeA(LPCSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

The size of the file in bytes.

Description

This function determines the size of the file specified by the file name.

1.1.88 FileGetSizeW Function

Retrieves the size of a specified file.

C++

```
DWORD WINAPI FileGetSizeW(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

The size of the file in bytes.

Description

This function determines the size of the file specified by the name of the file

1.1.89 FileIsExeA Function

Checks if the file is a Windows executable

C++

```
BOOL WINAPI FileIsExeA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows executable, otherwise returns FALSE.

1.1.90 FileIsExeW Function

Checks if the file is a Windows executable

C++

```
BOOL WINAPI FileIsExeW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows executable, otherwise returns FALSE.

1.1.91 FileIsIconA Function

Checks if the file is a Windows icon (.ico) file

C++

```
BOOL WINAPI FileIsIconA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows icon (.ico) file, otherwise returns FALSE.

1.1.92 FileIsIconW Function

Checks if the file is a Windows icon (.ico) file

C++

```
BOOL WINAPI FileIsIconW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is a Windows icon (.ico) file, otherwise returns FALSE.

1.1.93 FileIsImageA Function

Checks if the file is an image file

C++

```
BOOL WINAPI FileIsImageA(LPCSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

Returns TRUE if the file is an image file, otherwise returns FALSE.

1.1.94 FileIsImageW Function

Checks if the file is an image file

C++

```
BOOL WINAPI FileIsImageW(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

Returns TRUE if the file is an image file, otherwise returns FALSE.

1.1.95 FileIsLink Function

Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).

C++

```
BOOL WINAPI FileIsLink(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file to check

Returns

Returns TRUE if the file is a Microsoft Windows shortcut, otherwise returns FALSE

Description

This function is used to check if the file with provided file name is a Windows shortcut or not

1.1.96 FileOrFolderExistsA Function

Checks if the file or folder with the given name exists

C++

```
BOOL WINAPI FileOrFolderExistsA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The file or folder name

Returns

Returns TRUE if the file or folder exists, otherwise returns FALSE.

1.1.97 FileOrFolderExistsW Function

Checks if the file or folder with the given name exists

C++

```
BOOL WINAPI FileOrFolderExistsW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The file or folder name

Returns

Returns TRUE if the file or folder exists, otherwise returns FALSE.

1.1.98 FileRenameA Function

Renames the file

C++

```
BOOL WINAPI FileRenameA(LPSTR oldName, LPSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR oldName	File to be renamed
LPSTR newName	New name of the file

Returns

Returns TRUE if the file was successfully renamed, otherwise returns FALSE

1.1.99 FileRenameW Function

Renames the file

C++

```
BOOL WINAPI FileRenameW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR oldName	File to be renamed
LPWSTR newName	New name of the file

Returns

Returns TRUE if the file was successfully renamed, otherwise returns FALSE

1.1.100 FileWriteA Function

Writes string to the file

C++

```
void WINAPI FileWriteA(HANDLE handle, LPSTR text);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
LPSTR text	The text to write

1.1.101 FileWriteCharA Function

Writes one character to the file

C++

```
void WINAPI FileWriteCharA(HANDLE handle, CHAR text);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
CHAR text	The character to write

1.1.102 FileWriteCharW Function

Writes one character to the file

C++

```
void WINAPI FileWriteCharW(HANDLE handle, WCHAR text);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
WCHAR text	The character to write

1.1.103 FileWriteNewLineA Function

Writes new line sequence to the file

C++

```
void WINAPI FileWriteNewLineA(HANDLE handle);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

1.1.104 FileWriteNewLineW Function

Writes new line sequence to the file

C++

```
void WINAPI FileWriteNewLineW(HANDLE handle);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file

1.1.105 FileWriteW Function

Writes string to the file

C++

```
void WINAPI FileWriteW(HANDLE handle, LPWSTR text);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
HANDLE handle	The handle of the file
LPWSTR text	The text to write

1.1.106 fpreset Function

This is function fpreset.

C++

```
void fpreset();
```

File

System.h (see page 170)

1.1.107 FreeContainer Function

Deallocates ASIC container

C++

```
void WINAPI FreeContainer(LPVOID container);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function

Returns

None

Description

Call this function to deallocate container memory and release container handle.

1.1.108 FullPathA Function

Gets the full path to the file based on file name

C++

```
BOOL WINAPI FullPathA(LPSTR fileName, LPSTR fullName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR fullName	The full path to the file

1.1.109 FullPathW Function

Gets the full path to the file based on file name

C++

```
BOOL WINAPI FullPathW(LPWSTR fileName, LPWSTR fullName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR fullName	The full path to the file

1.1.110 GenerateAuthenticationSignatureA Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureA(LPSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.111 GenerateAuthenticationSignatureExA Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureExA(int readerNumber, LPSTR pinCode, BYTE*
dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.112 GenerateAuthenticationSignatureExW Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureExW(int readerNumber, LPWSTR pinCode, BYTE*
dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature

LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.113 GenerateAuthenticationSignatureW Function

Generate authentication signature

C++

```
BOOL WINAPI GenerateAuthenticationSignatureW(LPWSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate authentication signature using provided hash value

1.1.114 GenerateBMPA Function

Generates Windows Bitmap file with QR Code image

C++

```
void WINAPI GenerateBMPA(LPSTR fileName, LPSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generate Windows Bitmap file with encoded text as QR Code image

1.1.115 GenerateBMPW Function

Generates Windows Bitmap file with QR Code image

C++

```
void WINAPI GenerateBMPW(LPWSTR fileName, LPWSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generate Windows Bitmap file with encoded text as QR Code image

1.1.116 GenerateNonRepudiationSignatureA Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureA(LPSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.117 GenerateNonRepudiationSignatureExA Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureExA(int readerNumber, LPSTR pinCode, BYTE*
dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.118 GenerateNonRepudiationSignatureExW Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureExW(int readerNumber, LPWSTR pinCode, BYTE*
dataHash, int hashSize, BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.119 GenerateNonRepudiationSignatureW Function

Generate non repudiation signature

C++

```
BOOL WINAPI GenerateNonRepudiationSignatureW(LPWSTR pinCode, BYTE* dataHash, int hashSize,
BYTE* signature, LPDWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR pinCode	The card PIN code value
BYTE* dataHash	The hash value buffer
int hashSize	The size of the hash data buffer
BYTE* signature	The output buffer that contains the generated signature
LPDWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is successfully generated, otherwise returns FALSE

Description

Generate non repudiation signature using provided hash value

1.1.120 GeneratePNGA Function

Generates PNG file with QR Code image

C++

```
void WINAPI GeneratePNGA(LPSTR fileName, LPSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
LPSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generates PNG file with encoded text as QR Code image

1.1.121 GeneratePNGW Function

Generates PNG file with QR Code image

C++

```
void WINAPI GeneratePNGW(LPWSTR fileName, LPWSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Description

Generates PNG file with encoded text as QR Code image

1.1.122 GenerateQRCodeA Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.123 GenerateQRCodeExA Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.124 GenerateQRCodeExW Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.125 GenerateQRCodeW Function

Read eID card and save the identity information and address to PNG QR Code file

C++

```
BOOL WINAPI GenerateQRCodeW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and generate the QR Code PNG image

1.1.126 GetAllFiles Function

Returns the names of files in a specified directory.

C++

```
void WINAPI GetAllFiles(FOLDERENUMPROC lpEnumProc, LPWSTR folderName, LPWSTR searchMask,
LPARAM lParam);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
FOLDERENUMPROC lpEnumProc	Callback function
LPWSTR folderName	The name of the folder
LPWSTR searchMask	The search string to match against the names of files in path.
LPARAM lParam	Specifies an application-defined value to be passed to the callback function

Description

This function enumerates all files in the specified folder which names match the searchMask parameter and calls the callback function passing the name of the file to it.

1.1.127 GetCardBufferA Function

Gets XML or CSV information from the memory buffer

C++

```
void WINAPI GetCardBufferA(void* buffer, void* strDest, int count);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
void* buffer	Memory buffer with information
void* strDest	Destination buffer
int count	Destination buffer size

Returns

None

Description

Use this function to get the card information in CSV or XML format from the memory buffer

1.1.128 GetCardBufferSize Function

This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format

C++

```
int WINAPI GetCardBufferSize(void* buffer);
```

File

Swelio.h (see page 165)

Returns

The size of the XML or CSV buffer

Description

Use this function to get the size of the XML buffer before allocating the buffer in memory

1.1.129 GetCardBufferW Function

Gets XML or CSV information from the memory buffer

C++

```
void WINAPI GetCardBufferW(void* buffer, void* strDest, int count);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
void* buffer	Memory buffer with information
void* strDest	Destination buffer
int count	Destination buffer size

Returns

None

Description

Use this function to get the card information in CSV or XML format from the memory buffer

1.1.130 GetCardSerialNumber Function

Get the serial number of EID card

C++

```
BOOL WINAPI GetCardSerialNumber(int readerNumber, BYTE* serialNumber, LPDWORD
serialNumberSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
BYTE* serialNumber	The memory buffer for getting the card serial number
LPDWORD serialNumberSize	the size of the memory buffer in bytes

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to get the serial number of EID card into the memory buffer

1.1.131 GetCardVersion Function

Get the applet version number for card in the reader with specified number

C++

```
int WINAPI GetCardVersion(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The reader index, starting from 0

Returns

The card applet version number

Description

Get the version number of eid card applet using the zero-based reader index. The first reader has number 0, second - 1, etc... You can read the information only from one selected reader at once.

1.1.132 GetEncodedCertificateSize Function

Returns the size of the Base64 encoded certificate

C++

```
int WINAPI GetEncodedCertificateSize(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidCertificate certificate	The certificate data

Returns

Returns the size of the buffer needed to hold the encoded certificate

Description

Use this function to calculate the size of the buffer needed to encode the certificate

1.1.133 GetEncodedPhotoSize Function

Calculates buffer size for Base64 encoded photo

C++

```
int WINAPI GetEncodedPhotoSize(PeidPicture photo);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure

Returns

The desired size of the buffer

Description

Use this function to calculate the size of the buffer needed for Base64 encoding of the photo This can be useful for including the photo data to the text document, for example to XML file

1.1.134 GetFileMD5A Function

Gets the MD5 hash value for the file

C++

```
BOOL WINAPI GetFileMD5A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given file

1.1.135 GetFileMD5W Function

Gets the MD5 hash value for the file

C++

```
BOOL WINAPI GetFileMD5W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given file

1.1.136 GetFilesCountA Function

Calculates the number of files in the given folder

C++

```
int WINAPI GetFilesCountA(LPSTR folderName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR folderName	The name of the folder

Returns

The number of files in the given folder

1.1.137 GetFilesCountW Function

Calculates the number of files in the given folder

C++

```
int WINAPI GetFilesCountW(LPWSTR folderName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR folderName	The name of the folder

Returns

The number of files in the given folder

1.1.138 GetFileSHA1A Function

Gets the SHA1 hash value for the file

C++

```
BOOL WINAPI GetFileSHA1A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given file

1.1.139 GetFileSHA1W Function

Gets the SHA1 hash value for the file

C++

```
BOOL WINAPI GetFileSHA1W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given file

1.1.140 GetFileSHA256A Function

Gets the SHA256 hash value for the file

C++

```
BOOL WINAPI GetFileSHA256A(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given file

1.1.141 GetFileSHA256W Function

Gets the SHA256 hash value for the file

C++

```
BOOL WINAPI GetFileSHA256W(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The buffer to store the hash value
int bufferSize	The size of the buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given file

1.1.142 GetHBitmapA Function

Generates Windows Bitmap in memory with QR Code image

C++

```
HBITMAP WINAPI GetHBitmapA(LPSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Returns

The result of the function is HBITMAP handle. You have to destroy it by yourself when its not needed anymore.

Description

Generates Windows Bitmap in memory file with encoded text as QR Code image

1.1.143 GetHBitmapW Function

Generates Windows Bitmap in memory with QR Code image

C++

```
HBITMAP WINAPI GetHBitmapW(LPWSTR text, int margin, int size, int level);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPWSTR text	The text to encode
int margin	The margin from the border in points
int size	The size of the one point in pixels
int level	The error correction level

Returns

The result of the function is HBITMAP handle. You have to destroy it by yourself when its not needed anymore.

Description

Generates Windows Bitmap in memory file with encoded text as QR Code image

1.1.144 GetISOCodeA Function

Returns the country ISO code based on the nationality string

C++

```
BOOL WINAPI GetISOCodeA(LPCSTR nationality, LPSTR iso, int bufferSize);
```

File

NationalityConverter.h (see page 165)

Parameters

Parameters	Description
LPCSTR nationality	The nationality string
LPSTR iso	The ISO code memory buffer

int bufferSize	The size if the memory buffer

Returns

Returns TRUE if the ISO code is successfully obtained; FALSE otherwise

Description

This function converts the nationality string stored on ID card to the country ISO code

1.1.145 GetISOCodeW Function

Returns the country ISO code based on the nationality string

C++

```
BOOL WINAPI GetISOCodeW(LPCWSTR nationality, LPWSTR iso, int bufferSize);
```

File

NationalityConverter.h (see page 165)

Parameters

Parameters	Description
LPCWSTR nationality	The nationality string
LPWSTR iso	The ISO code memory buffer
int bufferSize	The size if the memory buffer

Returns

Returns TRUE if the ISO code is successfully obtained; FALSE otherwise

Description

This function converts the nationality string stored on ID card to the country ISO code

1.1.146 GetMD5 Function

Gets the MD5 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetMD5(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates MD5 hash value for the given memory buffer

1.1.147 GetPNGA Function

Writes PNG image to the memory buffer.

C++

```
void WINAPI GetPNGA(LPSTR text, int margin, int size, int level, LPINT bufSize,
__deref_opt_out void ** ppvBits);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPSTR text	The text to encode
int margin	The margin from the image border in points
int size	The size of the point in pixels
int level	The error correction level
LPINT bufSize	The size of the output buffer
__deref_opt_out void ** ppvBits	The buffer when the resulting image is stored

Description

Writes PNG image to the memory buffer. Can be useful for web development.

1.1.148 GetPNGW Function

Writes PNG image to the memory buffer.

C++

```
void WINAPI GetPNGW(LPWSTR text, int margin, int size, int level, LPINT bufSize,
__deref_opt_out void ** ppvBits);
```

File

quricol.h (see page 165)

Parameters

Parameters	Description
LPWSTR text	The text to encode
int margin	The margin from the image border in points
int size	The size of the point in pixels
int level	The error correction level
LPINT bufSize	The size of the output buffer
__deref_opt_out void ** ppvBits	The buffer when the resulting image is stored

Description

Writes PNG image to the memory buffer. Can be useful for web development.

1.1.149 GetReaderIndexA Function

Returns the zero-based reader index with specified name

C++

```
int WINAPI GetReaderIndexA(LPSTR readerName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR readerName	The name of the reader

Returns

The zero-based reader index

Description

Use this function to get the zero-based index of the card reader with specified name

1.1.150 GetReaderIndexW Function

Returns the zero-based reader index with specified name

C++

```
int WINAPI GetReaderIndexW(LPWSTR readerName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR readerName	The name of the reader

Returns

The zero-based reader index

Description

Use this function to get the zero-based index of the card reader with specified name

1.1.151 GetReaderNameA Function

Returns the name of the card reader

C++

```
int WINAPI GetReaderNameA(int readerNumber, LPSTR strDest, int count);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
LPSTR strDest	Destination string
int count	The number of characters to be copied

Returns

Returns the reader name length

Description

Returns the name of the card reader with the specified zero-based index

1.1.152 GetReaderNameLenA Function

Returns the length of the reader name

C++

```
int WINAPI GetReaderNameLenA(int readerNumber);
```

File

Swelio.h (see page 165)

Returns

The length of the reader name

Description

Returns the length of the reader name for the smart card reader with specified zero-based index

1.1.153 GetReaderNameLenW Function

Returns the length of the reader name

C++

```
int WINAPI GetReaderNameLenW(int readerNumber);
```

File

Swelio.h (see page 165)

Returns

The length of the reader name

Description

Returns the length of the reader name for the smart card reader with specified zero-based index

1.1.154 GetReaderNameW Function

Returns the name of the card reader

C++

```
int WINAPI GetReaderNameW(int readerNumber, LPWSTR strDest, int count);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
LPWSTR strDest	Destination string
int count	Number of characters to be copied

Returns

The character count of the reader name

Description

Returns the name of the card reader with the specified zero-based index

1.1.155 GetReadersCount Function

Get number of card readers connected to PC

C++

```
int WINAPI GetReadersCount(VOID);
```

File

Swelio.h (see page 165)

Returns

The number of the connected smart card readers

Description

Checks how many smart card readers are connected to PC. If there is no readers connected then the usage of the Swelio Engine is not possible. The engine can control the change of the number of the card readers and can raise an event when the reader is connected or disconnected from PC

1.1.156 GetSelectedReaderIndex Function

Returns the index of the active smart card reader

C++

```
int WINAPI GetSelectedReaderIndex();
```

File

Swelio.h (see page 165)

Returns

The index of the selected card reader. The first reader has index 0.

Description

The zero-based index of the selected card reader. If there is only one reader is connected to PC then this reader has the index 0 and it's a default (selected) reader.

1.1.157 GetSHA1 Function

Gets the SHA1 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetSHA1(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA1 hash value for the given memory buffer

1.1.158 GetSHA256 Function

Gets the SHA256 hash value for the content of the memory buffer

C++

```
BOOL WINAPI GetSHA256(BYTE* source, int sourceSize, BYTE* buffer, int bufferSize);
```

File

Encryption.h (see page 161)

Parameters

Parameters	Description
BYTE* source	The source memory block
int sourceSize	The size of the source memory block
BYTE* buffer	The buffer for the hash value
int bufferSize	The size of the destination buffer

Returns

The result of the function is equal to TRUE if operation is completed successfully, otherwise the result is FALSE

Description

Calculates SHA256 hash value for the given memory buffer

1.1.159 GetStartupA Function

Checks if the application is registered to run when Windows starts

C++

```
BOOL WINAPI GetStartupA(LPCSTR appName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCSTR appName	The name of the application

1.1.160 GetStartupW Function

Checks if the application is registered to run when Windows starts

C++

```
BOOL WINAPI GetStartupW(LPCWSTR appName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCWSTR appName	The name of the application

1.1.161 GetSupportSIS Function

Checks if the SIS cards are supported by the engine

C++

```
BOOL WINAPI GetSupportSIS();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE if SIS card support is activated, otherwise returns FALSE

Description

The SIS card reading operation takes more time than the reading of the eID card. By default when the card is inserted in the reader the engine will try to detect the card type and the card insertion event will be raised for eID cards only. If you want to support the SIS cards in your application then you have to activate it using SetSupportSIS (see page 120) function. Use GetSupportSIS function to check if the SIS card support is activated.

1.1.162 GetTextLineSize Function

This is function GetTextLineSize.

C++

```
SIZE WINAPI GetTextLineSize(LPCWSTR s, HFONT hFont);
```

File

Graphics.h (see page 164)

1.1.163 GetTextSize Function

This is function GetTextSize.

C++

```
SIZE WINAPI GetTextSize(LPCWSTR s, HFONT hFont, UINT flags);
```

File

Graphics.h (see page 164)

1.1.164 GetTextSizeEx Function

This is function GetTextSizeEx.

C++

```
SIZE WINAPI GetTextSizeEx(LPCWSTR s, HFONT hFont, UINT proposedWidth, UINT flags, BOOL
margins);
```

File

Graphics.h (see page 164)

1.1.165 HibernateWindows Function

Hibernates Windows

C++

```
BOOL WINAPI HibernateWindows();
```

File

System.h (see page 170)

1.1.166 InitializeContainer Function

Initializes ASIC container

C++

```
LPVOID WINAPI InitializeContainer();
```

File

Swelio.h (see page 165)

Returns

Retrns container handle pointer

Description

This functions initializes container handle needed for all container operations. Must be called first prior to other container-related calls

1.1.167 IsAnimatedGIFA Function

Checks if the file is an animated GIF image file

C++

```
BOOL WINAPI IsAnimatedGIFA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if the file is an animated GIF image file, otherwise returns FALSE.

1.1.168 IsAnimatedGIFW Function

Checks if the file is an animated GIF image file

C++

```
BOOL WINAPI IsAnimatedGIFW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if the file is an animated GIF image file, otherwise returns FALSE.

1.1.169 IsCardActivated Function

Checks the connection between a smart card and a reader

C++

```
BOOL WINAPI IsCardActivated();
```

File

Swelio.h (see page 165)

Description

This function checks the connection between the calling application and a smart card in the target reader.

1.1.170 IsCardActivatedEx Function

Checks the connection between a smart card and a reader

C++

```
BOOL WINAPI IsCardActivatedEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

This function checks the connection between the calling application and a smart card in the target reader.

1.1.171 IsCardPresent Function

Checks if the card is present in the card reader

C++

```
BOOL WINAPI IsCardPresent();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE if the card is inserted in the reader, otherwise returns FALSE

Description

Use IsCardPresent function to check if the card is inserted in the card reader or not

1.1.172 IsCardPresentEx Function

Checks if the card is present in the card reader

C++

```
BOOL WINAPI IsCardPresentEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Returns

Returns TRUE if the card is inserted in the reader, otherwise returns FALSE

Description

Use isCardPresent function to check if the card is inserted in the card reader or not

1.1.173 IsCardStillInserted Function

Checks if the card is still inserted in the card reader

C++

```
BOOL WINAPI IsCardStillInserted();
```

File

Swelio.h (see page 165)

Description

This function checks if the card is still present in the card reader

1.1.174 IsCardStillInsertedEx Function

Checks if the card is still inserted in the card reader

C++

```
BOOL WINAPI IsCardStillInsertedEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader

Description

This function checks if the card is still present in the card reader

1.1.175 IsCitrixSession Function

Checks if application is running in Citrix session

C++

```
BOOL WINAPI IsCitrixSession();
```

File

SystemInfo.h (see page 171)

1.1.176 IsConnectedToInternet Function

Checks if PC is connected to Internet

C++

```
BOOL WINAPI IsConnectedToInternet();
```

File

SystemInfo.h (see page 171)

1.1.177 IsDirectoryA Function

Verifies that a path is a valid directory.

C++

```
BOOL WINAPI IsDirectoryA(LPSTR folderName);
```

File

FileOperations.h (see page 162)

Returns

Returns TRUE if the path is a valid directory, or FALSE otherwise.

Description

This function verifies if provided value is the name of the folder

1.1.178 IsDirectoryW Function

Verifies that a path is a valid directory.

C++

```
BOOL WINAPI IsDirectoryW(LPWSTR folderName);
```

File

FileOperations.h (see page 162)

Returns

Returns TRUE if the path is a valid directory, or FALSE otherwise.

Description

This function verifies if provided value is the name of the folder

1.1.179 IsEIDCard Function

Check if Belgian EID card is inserted into card reader

C++

```
BOOL WINAPI IsEIDCard();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE, if Belgian eID card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.180 IsEIDCardEx Function

Check if Belgian EID card is inserted into card reader

C++

```
BOOL WINAPI IsEIDCardEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

Returns TRUE, if Belgian eID card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.181 IsEngineActive Function

Checks if the Swelio Engine is activated

C++

```
BOOL WINAPI IsEngineActive();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE if the Swelio Engine is active, otherwise returns FALSE.

Description

This function checks if the Engine already activated using the StartEngine (see page 121) function.

1.1.182 IsFemaleA Function

Checks if the card owner is female

C++

```
BOOL WINAPI IsFemaleA(PEidIdentityA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityA identity	The person identity information structure

Returns

Returns TRUE if the card owner is female, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.183 IsFemaleW Function

Checks if the card owner is female

C++

```
BOOL WINAPI IsFemaleW(PEidIdentityW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityW identity	The person identity information structure

Returns

Returns TRUE if the card owner is female, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.184 IsMaleA Function

Checks if the card owner is male

C++

```
BOOL WINAPI IsMaleA(PEidIdentityA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityA identity	The person identity information structure

Returns

Returns TRUE if the card owner is male, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.185 IsMaleW Function

Checks if the card owner is male

C++

```
BOOL WINAPI IsMaleW(PEidIdentityW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityW identity	The person identity information structure

Returns

Returns TRUE if the card owner is male, otherwise returns FALSE

Description

Use this function to check the gender of the card owner

1.1.186 IsMediaCenter Function

Checks if the Media Center version of Windows is installed

C++

```
BOOL WINAPI IsMediaCenter();
```

File

SystemInfo.h (see page 171)

1.1.187 IsMetroActive Function

Checks if metro interface is active

C++

```
BOOL WINAPI IsMetroActive();
```

File

SystemInfo.h (see page 171)

1.1.188 IsMultiTouchReady Function

Checks if the system is multi touch ready

C++

```
BOOL WINAPI IsMultiTouchReady();
```

File

SystemInfo.h (see page 171)

1.1.189 IsNativeWin64 Function

Checks if the application is native 64 bit executable

C++

```
BOOL WINAPI IsNativeWin64();
```

File

SystemInfo.h (see page 171)

1.1.190 IsRemoteSession Function

Checks if application is running in RDP session

C++

```
BOOL WINAPI IsRemoteSession();
```

File

SystemInfo.h (see page 171)

1.1.191 IsSISCard Function

Check if Belgian SIS card is inserted into card reader

C++

```
BOOL WINAPI IsSISCard();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE, if Belgian SIS card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.192 IsSISCardEx Function

Check if Belgian SIS card is inserted into card reader

C++

```
BOOL WINAPI IsSISCardEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

Returns TRUE, if Belgian SIS card is inserted in the reader. If there is no card in the reader or the card of other type is inserted, returns FALSE

Description

If the card is inserted in the reader, this function performs the card type check.

1.1.193 IsTabletPC Function

Checks if the application is running on the Tablet PC

C++

```
BOOL WINAPI IsTabletPC();
```

File

SystemInfo.h (see page 171)

1.1.194 IsUnicodeFileA Function

Checks if the file is UNICODE file

C++

```
BOOL WINAPI IsUnicodeFileA(LPCSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The name of the file

Returns

Returns TRUE if file is stored in UNICODE format, otherwise returns FALSE.

Description

This function checks the file encoding based on BOM (Byte Order Mark).

1.1.195 IsUnicodeFileW Function

Checks if the file is UNICODE file

C++

```
BOOL WINAPI IsUnicodeFileW(LPCWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The name of the file

Returns

Returns TRUE if file is stored in UNICODE format, otherwise returns FALSE.

Description

This function checks the file encoding based on BOM (Byte Order Mark).

1.1.196 IsValidFileNameA Function

Checks if provided string is a valid file name

C++

```
BOOL WINAPI IsValidFileNameA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file

Returns

Returns TRUE if provided string is valid file name, otherwise returns FALSE

Description

Checks if provided string is a valid file name and does not contain any illegal characters

1.1.197 IsValidFileNameW Function

Checks if provided string is a valid file name

C++

```
BOOL WINAPI IsValidFileNameW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file

Returns

Returns TRUE if provided string is valid file name, otherwise returns FALSE

Description

Checks if provided string is a valid file name and does not contain any illegal characters

1.1.198 IsValidPathNameA Function

Checks if provided string is a valid file path

C++

```
BOOL WINAPI IsValidPathNameA(LPSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The file path to check

Returns

Returns TRUE if provided string is valid file path, otherwise returns FALSE

Description

Checks if provided string is a valid file path and does not contain any illegal characters

1.1.199 IsValidPathNameW Function

Checks if provided string is a valid file path

C++

```
BOOL WINAPI IsValidPathNameW(LPWSTR fileName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The file path to check

Returns

Returns TRUE if provided string is valid file path, otherwise returns FALSE

Description

Checks if provided string is a valid file path and does not contain any illegal characters

1.1.200 IsWindows10 Function

Checks if PC is running Windows 10 or better

C++

```
BOOL WINAPI IsWindows10();
```

File

SystemInfo.h (see page 171)

1.1.201 IsWindows7 Function

Checks if PC is running Windows 7 or better

C++

```
BOOL WINAPI IsWindows7();
```

File

SystemInfo.h (see page 171)

1.1.202 IsWindows8 Function

Checks if PC is Running Windows 8 or better

C++

```
BOOL WINAPI IsWindows8();
```

File

SystemInfo.h (see page 171)

1.1.203 IsWindowsVista Function

Checks if PC is running Windows Vista or better

C++

```
BOOL WINAPI IsWindowsVista();
```

File

SystemInfo.h (see page 171)

1.1.204 IsWindowsXP Function

Checks if PC is running Windows XP

C++

```
BOOL WINAPI IsWindowsXP();
```

File

SystemInfo.h (see page 171)

1.1.205 IsWindowsXPSP2 Function

Checks if PC is running Windows XP with Service Pack 2 installed

C++

```
BOOL WINAPI IsWindowsXPSP2();
```

File

SystemInfo.h (see page 171)

1.1.206 IsWow64 Function

Checks if the 32 bit application runs on 64 bit Windows

C++

```
BOOL WINAPI IsWow64();
```

File

SystemInfo.h (see page 171)

1.1.207 LayeredWndProcA Function

The default window procedure for the layered window

C++

```
LRESULT CALLBACK LayeredWndProcA(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
```

File

System.h (see page 170)

1.1.208 LayeredWndProcW Function

The default window procedure for the layered window

C++

```
LRESULT CALLBACK LayeredWndProcW(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);
```

File

System.h (see page 170)

1.1.209 LoadBitmapJPG Function

This is function LoadBitmapJPG.

C++

```
HBITMAP WINAPI LoadBitmapJPG(LPCWSTR szFile, LPINT lpiWidth, LPINT lpiHeight,
__deref_opt_out void ** ppvBits);
```

File

Graphics.h (see page 164)

1.1.210 LoadBitmapPNG Function

This is function LoadBitmapPNG.

C++

```
HBITMAP WINAPI LoadBitmapPNG(LPCWSTR szFile, LPINT lpiWidth, LPINT lpiHeight,
__deref_opt_out void ** ppvBits);
```

File

Graphics.h (see page 164)

1.1.211 LoadCertificateA Function

Reads the certificate from a file

C++

```
void WINAPI LoadCertificateA(LPSTR fileName, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	The source file name
PEidCertificate certificate	The pointer to EidCertificate (see page 135) structure

Description

Use this function to read the certificate from the file

1.1.212 LoadCertificateW Function

Reads the certificate from a file

C++

```
void WINAPI LoadCertificateW(LPWSTR fileName, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	The source file name
PEidCertificate certificate	The pointer to EidCertificate (see page 135) structure

Description

Use this function to read the certificate from the file

1.1.213 LoadIdentityA Function

Reads the raw identity information from a file

C++

```
void WINAPI LoadIdentityA(LPSTR fileName, PEidIdentityA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	The name of the source file
PEidIdentityA identity	The pointer to EidIdentityA (see page 135) structure

Description

Use this function to read back the identity information stored to the file using SaveIdentityA (see page 107) function

1.1.214 LoadIdentityW Function

Reads the raw identity information from a file

C++

```
void WINAPI LoadIdentityW(LPWSTR fileName, PEidIdentityW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	The name of the source file
PEidIdentityW identity	The pointer to EidIdentityW (see page 137) structure

Description

Use this function to read back the identity information stored to the file using SaveIdentityW (see page 107) function

1.1.215 LoadPhotoA Function

Loads photo from a file

C++

```
void WINAPI LoadPhotoA(PeidPicture photo, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure
LPSTR fileName	Destination file name

Description

Loads raw picture data from a file

1.1.216 LoadPhotoW Function

Loads photo from a file

C++

```
void WINAPI LoadPhotoW(PeidPicture photo, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure
LPWSTR fileName	Destination file name

Description

Loads raw picture data from a file

1.1.217 LoadPNGResource Function

This is function LoadPNGResource.

C++

```
HBITMAP WINAPI LoadPNGResource(HMODULE handle, LPCWSTR szName, LPINT lpiWidth, LPINT
lpiHeight, __deref_opt_out void ** ppvBits);
```

File

Graphics.h (see page 164)

1.1.218 MakeCompatibleBitmap Function

This is function MakeCompatibleBitmap.

C++

```
HBITMAP WINAPI MakeCompatibleBitmap(HBITMAP source, int width, int height);
```

File

Graphics.h (see page 164)

1.1.219 MakeSoundFromFileA Function

Plays the wave sound from the file

C++

```
void WINAPI MakeSoundFromFileA(LPCSTR soundName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCSTR soundName	The name of the file

Description

This function plays a sound specified by the given file name.

1.1.220 MakeSoundFromFileW Function

Plays the wave sound from the file

C++

```
void WINAPI MakeSoundFromFileW(LPCWSTR soundName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCWSTR soundName	The name of the file

Description

This function plays a sound specified by the given file name.

1.1.221 MakeSoundFromResourceA Function

Plays the wave sound from the resource

C++

```
void WINAPI MakeSoundFromResourceA(HMODULE hModule, LPCSTR soundName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HMODULE hModule	Handle to the executable file that contains the resource to be loaded.
LPCSTR soundName	A string that specifies the sound to play.

Description

This function plays a sound specified by the given resource name.

1.1.222 MakeSoundFromResourceW Function

Plays the wave sound from the resource

C++

```
void WINAPI MakeSoundFromResourceW(HMODULE hModule, LPCWSTR soundName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HMODULE hModule	Handle to the executable file that contains the resource to be loaded.
LPCWSTR soundName	A string that specifies the sound to play.

Description

This function plays a sound specified by the given resource name.

1.1.223 PointsToPixels Function

This is function PointsToPixels.

C++

```
int WINAPI PointsToPixels(int points);
```

File

Graphics.h (see page 164)

1.1.224 PortAvailable Function

Checks if the port with specified number is available

C++

```
BOOL WINAPI PortAvailable(int port);
```

File

SystemInfo.h (see page 171)

1.1.225 ReadAddressA Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressA(PEidAddressA address);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidAddressA address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.226 ReadAddressExA Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressExA(int readerNumber, PEidAddressA address);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidAddressA address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.227 ReadAddressExW Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressExW(int readerNumber, PEidAddressW address);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidAddressW address	The pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.228 ReadAddressW Function

Read address information from Belgian eID card

C++

```
BOOL WINAPI ReadAddressW(PEidAddressW address);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidAddressW address	the pointer to the address information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.229 ReadAuthenticationCertificate Function

Read Authentication Certificate to memory

C++

```
BOOL WINAPI ReadAuthenticationCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Authentication Certificate from the card to EidCertificate (see page 135) structure

1.1.230 ReadAuthenticationCertificateEx Function

Read Authentication Certificate to memory

C++

```
BOOL WINAPI ReadAuthenticationCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Authentication Certificate from the card to EidCertificate (see page 135) structure

1.1.231 ReadBufferFromFileA Function

Reads the content of the file to the memory buffer

C++

```
void WINAPI ReadBufferFromFileA(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

Use this function to retrieve the content of the file to the memory block

1.1.232 ReadBufferFromFileW Function

Reads the content of the file to the memory buffer

C++

```
void WINAPI ReadBufferFromFileW(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

Use this function to retrieve the content of the file to the memory block

1.1.233 ReadCaCertificate Function

Read Ca Certificate to memory

C++

```
BOOL WINAPI ReadCaCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Ca Certificate to EidCertificate (see page 135) structure

1.1.234 ReadCaCertificateEx Function

Read Ca Certificate to memory

C++

```
BOOL WINAPI ReadCaCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Ca Certificate to EidCertificate (see page 135) structure

1.1.235 ReadIdentityA Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityA(PEidIdentityA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityA identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.236 ReadIdentityExA Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityExA(int readerNumber, PEidIdentityA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidIdentityA identity	The pointer to the identity information structure

Returns

Returns TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.237 ReadIdentityExW Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityExW(int readerNumber, PEidIdentityW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PEidIdentityW identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.238 ReadIdentityW Function

Read identity information from Belgian eID card

C++

```
BOOL WINAPI ReadIdentityW(PEidIdentityW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidIdentityW identity	The pointer to the identity information structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

1.1.239 ReadNonRepudiationCertificate Function

Read Non Repudiation Certificate to memory

C++

```
BOOL WINAPI ReadNonRepudiationCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Non Repudiation Certificate to EidCertificate (see page 135) structure

1.1.240 ReadNonRepudiationCertificateEx Function

Read Non Repudiation Certificate to memory

C++

```
BOOL WINAPI ReadNonRepudiationCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Non Repudiation Certificate to EidCertificate (see page 135) structure

1.1.241 ReadPhoto Function

Reads a photo from a card

C++

```
BOOL WINAPI ReadPhoto(PeidPicture photo);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

Description

Reads a photo from Belgian eID card to EidPicture (see page 138) structure. This structure holds the raw image bytes and the length of the image bytes array

1.1.242 ReadPhotoAsBitmap Function

Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
HBITMAP WINAPI ReadPhotoAsBitmap();
```

File

Swelio.h (see page 165)

Returns

A handle to a bitmap indicates success. NULL indicates failure.

1.1.243 ReadPhotoAsBitmapEx Function

Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the Windows bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
HBITMAP WINAPI ReadPhotoAsBitmapEx(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

Returns

A handle to a bitmap indicates success. NULL indicates failure.

1.1.244 ReadPhotoEx Function

Reads a photo from a card

C++

```
BOOL WINAPI ReadPhotoEx(int readerNumber, PeidPicture photo);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PeidPicture photo	The pointer to EidPicture (see page 138) structure

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

Description

Reads a photo from Belgian eID card to EidPicture (see page 138) structure

1.1.245 ReadRootCaCertificate Function

Read Root Ca Certificate to memory

C++

```
BOOL WINAPI ReadRootCaCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Root Ca Certificate to EidCertificate (see page 135) structure

1.1.246 ReadRootCaCertificateEx Function

Read Root Ca Certificate to memory

C++

```
BOOL WINAPI ReadRootCaCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Root Ca Certificate to EidCertificate (see page 135) structure

1.1.247 ReadRrnCertificate Function

Read Rrn Certificate to memory

C++

```
BOOL WINAPI ReadRrnCertificate(PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Rrn Certificate to EidCertificate (see page 135) structure

1.1.248 ReadRrnCertificateEx Function

Read Rrn Certificate to memory

C++

```
BOOL WINAPI ReadRrnCertificateEx(int readerNumber, PEidCertificate certificate);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader
cerificate	The pointer to EidCertificate (see page 135) structure

Returns

TRUE when certificate is successfully received from the card; otherwise returns FALSE

Description

Read Rrn Certificate to EidCertificate (see page 135) structure

1.1.249 ReadSISCardA Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardA(PSISRecordA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PSISRecordA	The pointer to SISRecordA (see page 144) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.250 ReadSISCardExA Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardExA(int readerNumber, PSISRecordA identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PSISRecordA	The pointer to SISRecordA (see page 144) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.251 ReadSISCardExW Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardExW(int readerNumber, PSISRecordW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
PSISRecordW	The pointer to SISRecordW (see page 145) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.252 ReadSISCardW Function

Read Belgian SIS card.

C++

```
BOOL WINAPI ReadSISCardW(PSISRecordW identity);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PSISRecordW	The pointer to SISRecordW (see page 145) structure

Returns

TRUE when information is successfully received from the card; otherwise returns FALSE

Description

Read the public information from the Belgian SIS card. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.1.253 RecycleBinEmpty Function

Returns TRUE if Windows Recyce Bin is empty

C++

```
BOOL WINAPI RecycleBinEmpty();
```

File

SystemInfo.h (see page 171)

1.1.254 ReloadReadersList Function

Reloads the list of the available card readers

C++

```
void WINAPI ReloadReadersList();
```

File

Swelio.h (see page 165)

Description

When the card reader is inserted or removed you may need to reload the list of the available card readers

1.1.255 RemoveCallback Function

Remove callback procedure for card events

C++

```
void WINAPI RemoveCallback();
```

File

Swelio.h (see page 165)

Description

Use this function to deactivate card events callback procedure

1.1.256 RemoveStartupA Function

Removes the application from the list of the automatically started applications

C++

```
void WINAPI RemoveStartupA(LPCSTR appName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCSTR appName	The name of the application

Description

For application that starts automatically when Windows starts removes it from the automatically launching applications list

1.1.257 RemoveStartupW Function

Removes the application from the list of the automatically started applications

C++

```
void WINAPI RemoveStartupW(LPCWSTR appName);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCWSTR appName	The name of the application

Description

For application that starts automatically when Windows starts removes it from the automatically launching applications list

1.1.258 RestoreWindowSubclassA Function

Restores window standard procedure

C++

```
void WINAPI RestoreWindowSubclassA(HWND hwnd);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND hwnd	The window handle

1.1.259 RestoreWindowSubclassW Function

Restores window standard procedure

C++

```
void WINAPI RestoreWindowSubclassW(HWND hwnd);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND hwnd	The window handle

1.1.260 SaveAuthenticationCertificateA Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Authentication Certificate from the card and save it to a file.

1.1.261 SaveAuthenticationCertificateExW Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Authentication Certificate from the card and save it to a file.

1.1.262 SaveAuthenticationCertificateW Function

Save Authentication Certificate to a file

C++

```
void WINAPI SaveAuthenticationCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Authentication Certificate from the card and save it to a file.

1.1.263 SaveCaCertificateA Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Ca Certificate from the card and save it to a file

1.1.264 SaveCaCertificateExW Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Ca Certificate from the card and save it to a file

1.1.265 SaveCaCertificateW Function

Save Ca Certificate to a file

C++

```
void WINAPI SaveCaCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Ca Certificate from the card and save it to a file

1.1.266 SaveCardToToXMLStreamExA Function

Read eID card and save the information to XML buffer

C++

```
BOOL WINAPI SaveCardToToXMLStreamExA(int readerNumber, void* buffer);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
void* buffer	The memory buffer to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML buffer in the memory.

1.1.267 SaveCardToToXMLStreamExW Function

Read eID card and save the information to XML buffer

C++

```
BOOL WINAPI SaveCardToToXMLStreamExW(int readerNumber, void* buffer);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
void* buffer	The memory buffer to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML buffer in the memory.

1.1.268 SaveCardToXmlA Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.269 SaveCardToXmlExA Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.270 SaveCardToXmlExW Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.271 SaveCardToXmlW Function

Read eID card and save the information to XML file

C++

```
BOOL WINAPI SaveCardToXmlW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to XML file.

1.1.272 SaveContainer Function

Save container to the file

C++

```
BOOL WINAPI SaveContainer(LPVOID container, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPVOID container	Container handle, which is allocated by calling InitializeContainer (see page 70) function
LPWSTR fileName	Desired name of the container file

Returns

Returns true if the operation is successful, otherwise returns false

Description

After adding all necessary files to the container call this function to save container to the file

1.1.273 SaveIdentityA Function

Saves indentity information to a file

C++

```
void WINAPI SaveIdentityA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	The name of the destination file

Description

Use this function to store the raw identity information from the Belgian eID card to a file. You can use LoadIdentityA (see page 85) to read this information from the file to EidIdentityA (see page 135) structure

1.1.274 SaveIdentityW Function

Saves indentity information to a file

C++

```
void WINAPI SaveIdentityW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	The name of the destination file

Description

Use this function to store the raw identity information from the Belgian eID card to a file. You can use LoadIdentityW (see page 85) to read this information from the file to EidIdentityW (see page 137) structure

1.1.275 SaveNonRepudiationCertificateA Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.276 SaveNonRepudiationCertificateExW Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.277 SaveNonRepudiationCertificateW Function

Save Non Repudiation Certificate to a file

C++

```
void WINAPI SaveNonRepudiationCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Non Repudiation Certificate from the card and save it to a file

1.1.278 SavePersonCsvToStreamA Function

Read eID card and save the identity information to CSV memory buffer

C++

```
BOOL WINAPI SavePersonCsvToStreamA(int readerNumber, void* buffer);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	the card reader index
void* buffer	Memory buffer

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV memory buffer.

1.1.279 SavePersonCsvToStreamW Function

Read eID card and save the identity information to CSV memory buffer

C++

```
BOOL WINAPI SavePersonCsvToStreamW(int readerNumber, void* buffer);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	the card reader index
void* buffer	Memory buffer

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV memory buffer.

1.1.280 SavePersonToCsvA Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.281 SavePersonToCsvExA Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.282 SavePersonToCsvExW Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.283 SavePersonToCsvW Function

Read eID card and save the identity information and address to CSV file

C++

```
BOOL WINAPI SavePersonToCsvW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store information

Returns

Returns TRUE if the information is retrieved from the card, otherwise returns FALSE

Description

Use this function to read the information about the owner of the card and save it to CSV file.

1.1.284 SavePhotoA Function

Save photo to a file

C++

```
void WINAPI SavePhotoA(PeidPicture photo, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure
LPSTR fileName	Destination file name

Description

Save the raw picture data to a file

1.1.285 SavePhotoAsBitmapA Function

Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.286 SavePhotoAsBitmapExA Function

Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.287 SavePhotoAsBitmapExW Function

Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	the zero-based index of the card reader.
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.288 SavePhotoAsBitmapW Function

Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsBitmapW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.289 SavePhotoAsJpegA Function

Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.290 SavePhotoAsJpegExA Function

Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegExA(int readerNumber, LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	the zero-based index of the card reader.
LPSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.291 SavePhotoAsJpegExW Function

Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegExW(int readerNumber, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.292 SavePhotoAsJpegW Function

Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.

C++

```
BOOL WINAPI SavePhotoAsJpegW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the photo

Returns

Returns TRUE if the photo is retrieved from the card, otherwise return FALSE

1.1.293 SavePhotoW Function

Saves photo to a file

C++

```
void WINAPI SavePhotoW(PeidPicture photo, LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PeidPicture photo	The pointer to EidPicture (see page 138) structure
LPWSTR fileName	Destination file name

Description

Saves the raw picture data to a file

1.1.294 SaveRootCaCertificateA Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read Root CA certificate from the card and save it to a file

1.1.295 SaveRootCaCertificateExW Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read Root CA certificate from the card and save it to a file

1.1.296 SaveRootCaCertificateW Function

Save Root Ca Certificate to a file

C++

```
void WINAPI SaveRootCaCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read Root CA certificate from the card and save it to a file

1.1.297 SaveRrnCertificateA Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateA(LPSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR fileName	File name to store the certificate

Description

Read RRN certificate from the card and save it to a file

1.1.298 SaveRrnCertificateExW Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateExW(LPWSTR fileName, int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate
int readerNumber	The reader index, starting from 0

Description

Read RRN certificate from the card and save it to a file

1.1.299 SaveRrnCertificateW Function

Save RRN Certificate to a file

C++

```
void WINAPI SaveRrnCertificateW(LPWSTR fileName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR fileName	File name to store the certificate

Description

Read RRN certificate from the card and save it to a file

1.1.300 SelectReader Function

When more than 1 reader connected, select the reader with specified number

C++

```
BOOL WINAPI SelectReader(int readerNumber);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The reader index, starting from 0

Returns

TRUE if the reader is selected, FALSE if the reader with specified number does not exist

Description

Selects the default card reader using the zero-based reader index. The first reader has number 0, second - 1, etc... You can read the information only from one selected reader at once.

1.1.301 SelectReaderByNameA Function

Select active smart card reader by providing the reader name

C++

```
BOOL WINAPI SelectReaderByNameA(LPSTR readerName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR readerName	The name of the card reader

Returns

TRUE if the reader is selected. If the reader with specified name is not found - returns FALSE

Description

Activates the reader with specified name

1.1.302 SelectReaderByNameW Function

Select active smart card reader by providing the reader name

C++

```
BOOL WINAPI SelectReaderByNameW(LPWSTR readerName);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR readerName	The name of the card reader

Returns

Returns TRUE if the reader is selected. If the reader with specified name is not found - returns FALSE

Description

Activates the reader with specified name

1.1.303 SendAPDU Function

This is function SendAPDU.

C++

```
BOOL WINAPI SendAPDU(int readerNumber, LPCBYTE apdu, DWORD apduLen, PCHAR result, LPDWORD
len);
```

File

Swelio.h (see page 165)

1.1.304 SetCallback Function

Activates callback procedure for card status change event

C++

```
void WINAPI SetCallback(CALLBACK_HANDLER callback, LPVOID userContext);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
CALLBACK_HANDLER callback	The pointer to callback procedure
LPVOID userContext	The user defined value passed to the callback procedure

Description

Your application can be notified about insertion or removal of the card from the card reader and the changes of the available card readers list (the reader is connected or disconnected from PC) Use this function to install the callback procedure

1.1.305 SetMWCompatibility Function

Set the compatibility mode with the old version of the oficial EID MiddleWare

C++

```
void WINAPI SetMWCompatibility();
```

File

Swelio.h (see page 165)

Description

The compatibility mode can be useful when the MiddleWare version 1.x or 2.x is installed on the target PC. Usually the more recent MiddleWare is used and this function is provided for backward compatibility only

1.1.306 SetStartupA Function

Register application to run when Windows starts

C++

```
void WINAPI SetStartupA(LPCSTR appName, LPCWSTR appPath);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCSTR appName	The name of the application
LPCWSTR appPath	The path to the application executable

1.1.307 SetStartupW Function

Register application to run when Windows starts

C++

```
void WINAPI SetStartupW(LPCWSTR appName, LPCWSTR appPath);
```

File

System.h (see page 170)

Parameters

Parameters	Description
LPCWSTR appName	The name of the application
LPCWSTR appPath	The path to the application executable

1.1.308 SetSupportSIS Function

Activates or deactivates SIS card support by engine

C++

```
void WINAPI SetSupportSIS(BOOL value);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
BOOL value	The SIS card support status

Description

Use SetSupportSIS to activate or deactivate the SIS card detection and reading. Even if SIS card support is activated it can be used only with ACR38U card readers Other card readers are not supported.

1.1.309 ShellCopyFileA Function

Copies file to the new location

C++

```
void WINAPI ShellCopyFileA(LPSTR oldName, LPSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR oldName	The source file name
LPSTR newName	The destination file name

Description

Copies file to the new location using Windows shell copy routine

1.1.310 ShellCopyFileW Function

Copies file to the new location

C++

```
void WINAPI ShellCopyFileW(LPWSTR oldName, LPWSTR newName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR oldName	The source file name
LPWSTR newName	The destination file name

Description

Copies file to the new location using Windows shell copy routine

1.1.311 ShutdownWindows Function

Logs off the interactive user, shuts down the system.

C++

```
BOOL WINAPI ShutdownWindows(UINT flags);
```

File

System.h (see page 170)

Returns

If the function succeeds returns TRUE, otherwise returns FALSE

Description

Logs off the interactive user, shuts down the system, or shuts down and restarts the system. It sends the WM_QUERYENDSESSION message to all applications to determine if they can be terminated.

This function accepts the following parameter:

flags : The shutdown type. This parameter must include one of the following values:

Value	Meaning
EWX_LOGOFF	Shuts down all processes running in the logon session of the process that called the ExitWindowsEx function. Then it logs the user off.
EWX_POWEROFF	Shuts down the system and turns off the power. The system must support the power-off feature.
EWX_REBOOT	Shuts down the system and then restarts the system.
EWX_RESTARTAPPS	Shuts down the system and then restarts it
EWX_SHUTDOWN	Shuts down the system to a point at which it is safe to turn off the power.

1.1.312 StartEngine Function

Activates the Swelio Engine.

C++

```
BOOL WINAPI StartEngine();
```

File

Swelio.h (see page 165)

Returns

Returns TRUE if the Swelio Engine is successfully started; otherwize returns FALSE

Description

This procedure must be called first before any other functions from Swelio library can be used.

1.1.313 StopEngine Function

Deactivates the Swelio Engine

C++

```
void WINAPI StopEngine();
```

File

Swelio.h (see page 165)

Description

Deactivates the Swelio Engine and clean up the used memory. Call this procedure at the end of you application once to finalize the usage of the Swelio Engine.

1.1.314 StretchNativeBitmap Function

This is function StretchNativeBitmap.

C++

```
BOOL WINAPI StretchNativeBitmap(HBITMAP src, HBITMAP dst, int srcWidth, int srcHeight, int dstWidth, int dstHeight);
```

File

Graphics.h (see page 164)

1.1.315 StripFileNameA Function

Replaces environment variable names with values

C++

```
void WINAPI StripFileNameA(LPCSTR fileName, LPSTR fullName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCSTR fileName	The source file name
LPSTR fullName	The expanded file name

Description

This function expands environment-variable strings and replaces them with their defined values in the file name.

1.1.316 StripFileNameW Function

Replaces environment variable names with values

C++

```
void WINAPI StripFileNameW(LPCWSTR fileName, LPWSTR fullName);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPCWSTR fileName	The source file name
LPWSTR fullName	The expanded file name

Description

This function expands environment-variable strings and replaces them with their defined values in the file name.

1.1.317 SuspendWindows Function

Suspends Windows

C++

```
BOOL WINAPI SuspendWindows();
```

File

System.h (see page 170)

1.1.318 TurnMonitorOff Function

Turns the monitor off

C++

```
void WINAPI TurnMonitorOff();
```

File

System.h (see page 170)

1.1.319 TurnMonitorOn Function

Turns the monitor on

C++

```
void WINAPI TurnMonitorOn();
```

File

System.h (see page 170)

1.1.320 UpdateWindowPosition Function

Updated the window position

C++

```
void WINAPI UpdateWindowPosition(HWND handle, int x, int y);
```

File

System.h (see page 170)

Parameters

Parameters	Description
HWND handle	The handle of the window
int x	New horizontal coordinate
int y	New vertical coordinate

1.1.321 VerifyPinA Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinA(LPSTR value);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.322 VerifyPinExA Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinExA(int readerNumber, LPSTR value);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.

LPSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.323 VerifyPinExW Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinExW(int readerNumber, LPWSTR value);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
int readerNumber	The zero-based index of the card reader.
LPWSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.324 VerifyPinW Function

Verify PIN code

C++

```
BOOL WINAPI VerifyPinW(LPWSTR value);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
LPWSTR value	PIN code to verify

Returns

TRUE when the correct PIN code is provided; otherwise returns FALSE

1.1.325 VerifySignature Function

Verifies the signature from the specified hash value.

C++

```
BOOL WINAPI VerifySignature(PEidCertificate certificate, BYTE* buffer, int bufferSize,
BYTE* signature, DWORD signatureSize);
```

File

Swelio.h (see page 165)

Parameters

Parameters	Description
PEidCertificate certificate	The public certificate
BYTE* buffer	The hash buffer
int bufferSize	The size of the hash buffer
BYTE* signature	The signature to be verified.
DWORD signatureSize	The size of the signature buffer

Returns

Returns TRUE if the signature is valid for the hash; otherwise, FALSE.

Description

Verify the signature using the public certificate of the signer

1.1.326 WriteBufferToFileA Function

Writes the memory buffer to file

C++

```
void WINAPI WriteBufferToFileA(LPSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

This function stores the content of the memory buffer to the file.

1.1.327 WriteBufferToFileW Function

Writes the memory buffer to file

C++

```
void WINAPI WriteBufferToFileW(LPWSTR fileName, BYTE* buffer, int bufferSize);
```

File

FileOperations.h (see page 162)

Parameters

Parameters	Description
LPWSTR fileName	The name of the file
BYTE* buffer	The address of the memory block
int bufferSize	The size of the memory block

Description

This function stores the content of the memory buffer to the file.

1.2 Structs, Records, Enums

The following table lists structs, records, enums in this documentation.

Enumerations

	Name	Description
	tagCardEventType (see page 128)	The type of the reader event
	CardEventType (see page 134)	The type of the reader event

Structures

	Name	Description
	tagEidAddressA (see page 128)	EID address information, stored on the card - ANSI version
	tagEidAddressW (see page 128)	EID address information, stored on the card - UNICODE version
	tagEidCertificate (see page 129)	Certificate, stored on EID card
	tagEidIdentityA (see page 129)	Identity information stored on EID card - ANSI version
	tagEidIdentityW (see page 131)	Identity information stored on EID card - UNICODE version
	tagEidPicture (see page 132)	Raw picture data from EID card
	tagSISRecordA (see page 132)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	tagSISRecordW (see page 133)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	EidAddressA (see page 134)	EID address information, stored on the card - ANSI version
	EidAddressW (see page 135)	EID address information, stored on the card - UNICODE version
	EidCertificate (see page 135)	Certificate, stored on EID card
	EidIdentityA (see page 135)	Identity information stored on EID card - ANSI version
	EidIdentityW (see page 137)	Identity information stored on EID card - UNICODE version
	EidPicture (see page 138)	Raw picture data from EID card
	PEidAddressA (see page 139)	EID address information, stored on the card - ANSI version
	PEidAddressW (see page 139)	EID address information, stored on the card - UNICODE version
	PEidCertificate (see page 139)	Certificate, stored on EID card
	PEidIdentityA (see page 140)	Identity information stored on EID card - ANSI version
	PEidIdentityW (see page 141)	Identity information stored on EID card - UNICODE version
	PeidPicture (see page 143)	Raw picture data from EID card
	PSISRecordA (see page 143)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	PSISRecordW (see page 144)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordA (see page 144)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

	SISRecordW (see page 145)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.2.1 tagCardEventType Enumeration

The type of the reader event

C++

```
enum tagCardEventType {
  ewtUnknownEvent,
  ewtCardInsert,
  ewtCardRemove,
  ewtReadersChange
};
```

File

CardEvents.h (see page 159)

Members

Members	Description
ewtUnknownEvent	Unknown event
ewtCardInsert	The card was inserted in the reader
ewtCardRemove	The card was removed from the reader
ewtReadersChange	The readers list changed

1.2.2 tagEidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
struct tagEidAddressA {
  char street[EID_MAX_STREET_LEN+1];
  char zip[EID_MAX_ZIP_LEN+1];
  char municipality[EID_MAX_MUNICIPALITY_LEN+1];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name
char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.3 tagEidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
struct tagEidAddressW {
  WCHAR street[EID_MAX_STREET_LEN+1];
  WCHAR zip[EID_MAX_ZIP_LEN+1];
  WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.4 tagEidCertificate Structure

Certificate, stored on EID card

C++

```
struct tagEidCertificate {
  BYTE certificate[EID_MAX_CERT_LEN+1];
  int certificateLength;
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.5 tagEidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
struct tagEidIdentityA {
  char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
  char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
  char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
  char validityDateEnd[EID_MAX_DATE_END_LEN +1];
  char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
  char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
  char name[EID_MAX_NAME_LEN+1];
  char firstName1[EID_MAX_FIRST_NAME1_LEN+1];
  char firstName2[EID_MAX_FIRST_NAME2_LEN+1];
  char nationality[EID_MAX_NATIONALITY_LEN+1];
  char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
  char birthDate[EID_MAX_BIRTHDATE_LEN+1];
  char sex[EID_MAX_SEX_LEN+1];
  char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
  int documentType;
  BOOL whiteCane;
```

```
    BOOL yellowCane;
    BOOL extendedMinority;
    char duplicate[EID_MAX_DUPLICATE_LEN + 1];
    char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
    BOOL memberOfFamily;
    char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
    char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
    char vat1[EID_MAX_VAT1_LEN + 1];
    char vat2[EID_MAX_VAT2_LEN + 1];
    char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
    char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
    char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority
char duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work Permit type
char vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
char vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number

char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.6 tagEidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```
struct tagEidIdentityW {
  WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
  WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
  WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
  WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
  WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
  WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
  WCHAR name[EID_MAX_NAME_LEN+1];
  WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];
  WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
  WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
  WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
  WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
  WCHAR sex[EID_MAX_SEX_LEN+1];
  WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
  int documentType;
  BOOL whiteCane;
  BOOL yellowCane;
  BOOL extendedMinority;
  WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];
  WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
  BOOL memberOfFamily;
  WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
  WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
  WCHAR vat1[EID_MAX_VAT1_LEN + 1];
  WCHAR vat2[EID_MAX_VAT2_LEN + 1];
  WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
  WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
  WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality

WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority
WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work permit type
WCHAR vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
WCHAR vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number
WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.7 tagEidPicture Structure

Raw picture data from EID card

C++

```
struct tagEidPicture {
  BYTE picture[EID_MAX_PICTURE_LEN+1];
  int pictureLength;
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.8 tagSISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
struct tagSISRecordA {
  char Name[SIS_MAX_NAME_LEN + 1];
  char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  char Initial[SIS_MAX_INITIAL_LEN+ 1];
  char Sex[SIS_MAX_SEX_LEN + 1];
  char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  char CardName[SIS_MAX_CARDNAME_LEN +1 ];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
char Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.9 tagSISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
struct tagSISRecordW {
  WCHAR Name[SIS_MAX_NAME_LEN + 1];
  WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
  WCHAR Sex[SIS_MAX_SEX_LEN + 1];
  WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  WCHAR CardName[SIS_MAX_CARDNAME_LEN +1 ];
};
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.10 CardEventType Enumeration

The type of the reader event

C++

```
typedef enum tagCardEventType {
  ewtUnknownEvent,
  ewtCardInsert,
  ewtCardRemove,
  ewtReadersChange
} CardEventType;
```

File

CardEvents.h (see page 159)

Members

Members	Description
ewtUnknownEvent	Unknown event
ewtCardInsert	The card was inserted in the reader
ewtCardRemove	The card was removed from the reader
ewtReadersChange	The readers list changed

1.2.11 EidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
typedef struct tagEidAddressA {
  char street[EID_MAX_STREET_LEN+1];
  char zip[EID_MAX_ZIP_LEN+1];
  char municipality[EID_MAX_MUNICIPALITY_LEN+1];
} EidAddressA, * PEidAddressA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name
char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.12 EidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
typedef struct tagEidAddressW {
  WCHAR street[EID_MAX_STREET_LEN+1];
  WCHAR zip[EID_MAX_ZIP_LEN+1];
  WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];
} EidAddressW, * PEidAddressW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.13 EidCertificate Structure

Certificate, stored on EID card

C++

```
typedef struct tagEidCertificate {
  BYTE certificate[EID_MAX_CERT_LEN+1];
  int certificateLength;
} EidCertificate, * PEidCertificate;
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.14 EidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
typedef struct tagEidIdentityA {
```

```
    char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
    char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
    char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
    char validityDateEnd[EID_MAX_DATE_END_LEN +1];
    char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
    char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
    char name[EID_MAX_NAME_LEN+1];
    char firstName1[EID_MAX_FIRST_NAME1_LEN+1];
    char firstName2[EID_MAX_FIRST_NAME2_LEN+1];
    char nationality[EID_MAX_NATIONALITY_LEN+1];
    char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
    char birthDate[EID_MAX_BIRTHDATE_LEN+1];
    char sex[EID_MAX_SEX_LEN+1];
    char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
    int documentType;
    BOOL whiteCane;
    BOOL yellowCane;
    BOOL extendedMinority;
    char duplicate[EID_MAX_DUPLICATE_LEN + 1];
    char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
    BOOL memberOfFamily;
    char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
    char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
    char vat1[EID_MAX_VAT1_LEN + 1];
    char vat2[EID_MAX_VAT2_LEN + 1];
    char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
    char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
    char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
} EidIdentityA, * PEidIdentityA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority

char duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work Permit type
char vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
char vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number
char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.15 EidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```
typedef struct tagEidIdentityW {
  WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
  WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
  WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
  WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
  WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
  WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
  WCHAR name[EID_MAX_NAME_LEN+1];
  WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];
  WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
  WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
  WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
  WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
  WCHAR sex[EID_MAX_SEX_LEN+1];
  WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
  int documentType;
  BOOL whiteCane;
  BOOL yellowCane;
  BOOL extendedMinority;
  WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];
  WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
  BOOL memberOfFamily;
  WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
  WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
  WCHAR vat1[EID_MAX_VAT1_LEN + 1];
  WCHAR vat2[EID_MAX_VAT2_LEN + 1];
  WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
  WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
  WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
} EidIdentityW, * PEidIdentityW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number

WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority
WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work permit type
WCHAR vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
WCHAR vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number
WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.16 EidPicture Structure

Raw picture data from EID card

C++

```
typedef struct tagEidPicture {
  BYTE picture[EID_MAX_PICTURE_LEN+1];
  int pictureLength;
} EidPicture, * PeidPicture;
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.17 PEidAddressA Structure

EID address information, stored on the card - ANSI version

C++

```
typedef struct tagEidAddressA {
  char street[EID_MAX_STREET_LEN+1];
  char zip[EID_MAX_ZIP_LEN+1];
  char municipality[EID_MAX_MUNICIPALITY_LEN+1];
} EidAddressA, * PEidAddressA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char street[EID_MAX_STREET_LEN+1];	Street name
char zip[EID_MAX_ZIP_LEN+1];	ZIP code
char municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.18 PEidAddressW Structure

EID address information, stored on the card - UNICODE version

C++

```
typedef struct tagEidAddressW {
  WCHAR street[EID_MAX_STREET_LEN+1];
  WCHAR zip[EID_MAX_ZIP_LEN+1];
  WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];
} EidAddressW, * PEidAddressW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR street[EID_MAX_STREET_LEN+1];	Street name
WCHAR zip[EID_MAX_ZIP_LEN+1];	ZIP code
WCHAR municipality[EID_MAX_MUNICIPALITY_LEN+1];	Municipality

1.2.19 PEidCertificate Structure

Certificate, stored on EID card

C++

```
typedef struct tagEidCertificate {
  BYTE certificate[EID_MAX_CERT_LEN+1];
  int certificateLength;
} EidCertificate, * PEidCertificate;
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE certificate[EID_MAX_CERT_LEN+1];	Certificate raw data buffer
int certificateLength;	Certificate data length

1.2.20 PEidIdentityA Structure

Identity information stored on EID card - ANSI version

C++

```
typedef struct tagEidIdentityA {
  char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
  char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
  char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
  char validityDateEnd[EID_MAX_DATE_END_LEN +1];
  char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
  char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
  char name[EID_MAX_NAME_LEN+1];
  char firstName1[EID_MAX_FIRST_NAME1_LEN+1];
  char firstName2[EID_MAX_FIRST_NAME2_LEN+1];
  char nationality[EID_MAX_NATIONALITY_LEN+1];
  char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
  char birthDate[EID_MAX_BIRTHDATE_LEN+1];
  char sex[EID_MAX_SEX_LEN+1];
  char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
  int documentType;
  BOOL whiteCane;
  BOOL yellowCane;
  BOOL extendedMinority;
  char duplicate[EID_MAX_DUPLICATE_LEN + 1];
  char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
  BOOL memberOfFamily;
  char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
  char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
  char vat1[EID_MAX_VAT1_LEN + 1];
  char vat2[EID_MAX_VAT2_LEN + 1];
  char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
  char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
  char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
} EidIdentityA, * PEidIdentityA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
char chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
char validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
char validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date

char municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
char nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
char name[EID_MAX_NAME_LEN+1];	Surname
char firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
char firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
char nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
char birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
char birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
char sex[EID_MAX_SEX_LEN+1];	Sex
char nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority
char duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
char specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
char dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
char workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work Permit type
char vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
char vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
char regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number
char brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
char brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.21 PEidIdentityW Structure

Identity information stored on EID card - UNICODE version

C++

```
typedef struct tagEidIdentityW {
  WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];
  WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];
  WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];
  WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];
  WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];
  WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];
  WCHAR name[EID_MAX_NAME_LEN+1];
  WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];
  WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];
  WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];
  WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];
  WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];
  WCHAR sex[EID_MAX_SEX_LEN+1];
```

```
  WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];
  int documentType;
  BOOL whiteCane;
  BOOL yellowCane;
  BOOL extendedMinority;
  WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];
  WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];
  BOOL memberOfFamily;
  WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];
  WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];
  WCHAR vat1[EID_MAX_VAT1_LEN + 1];
  WCHAR vat2[EID_MAX_VAT2_LEN + 1];
  WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];
  WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];
  WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];
} EidIdentityW, * PEidIdentityW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR cardNumber[EID_MAX_CARD_NUMBER_LEN+1];	Electronic ID card number
WCHAR chipNumber[EID_MAX_CHIP_NUMBER_LEN+1];	Electronic ID card physical chip number
WCHAR validityDateBegin[EID_MAX_DATE_BEGIN_LEN+1];	Card validity start date
WCHAR validityDateEnd[EID_MAX_DATE_END_LEN +1];	Card validity end date
WCHAR municipality[EID_MAX_DELIVERY_MUNICIPALITY_LEN+1];	Card delivery municipality
WCHAR nationalNumber[EID_MAX_NATIONAL_NUMBER_LEN+1];	National number
WCHAR name[EID_MAX_NAME_LEN+1];	Surname
WCHAR firstName1[EID_MAX_FIRST_NAME1_LEN+1];	First name (2 first given names)
WCHAR firstName2[EID_MAX_FIRST_NAME2_LEN+1];	First name part 2 (first letter of the 3rd given name).
WCHAR nationality[EID_MAX_NATIONALITY_LEN+1];	Nationality
WCHAR birthLocation[EID_MAX_BIRTHPLACE_LEN+1];	Birth location
WCHAR birthDate[EID_MAX_BIRTHDATE_LEN+1];	Birth date
WCHAR sex[EID_MAX_SEX_LEN+1];	Sex
WCHAR nobleCondition[EID_MAX_NOBLE_CONDITION_LEN+1];	Noble condition
int documentType;	Document type code (Belgian citizen card, Kids Card, Foreigner card)
BOOL whiteCane;	White cane (blind people)
BOOL yellowCane;	Yellow cane (partially sighted people)
BOOL extendedMinority;	Extended minority
WCHAR duplicate[EID_MAX_DUPLICATE_LEN + 1];	Duplicata
WCHAR specialOrganization[EID_MAX_SPECIALORGANIZATION_LEN + 1];	Special Organization
BOOL memberOfFamily;	Member of family
WCHAR dateAndCountryOfProtection[EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN + 1];	Date and country of protection
WCHAR workPermitType[EID_MAX_WORKPERMITTYPE_LEN + 1];	Work permit type

WCHAR vat1[EID_MAX_VAT1_LEN + 1];	Employer VAT1
WCHAR vat2[EID_MAX_VAT2_LEN + 1];	Employer VAT2
WCHAR regionalFileNumber[EID_MAX_REGIONALFILENUMBER_LEN + 1];	Regional file number
WCHAR brexitMention1[EID_MAX_BREXITMENTION1_LEN + 1];	BREXIT
WCHAR brexitMention2[EID_MAX_BREXITMENTION2_LEN + 1];	BREXIT

1.2.22 PeidPicture Structure

Raw picture data from EID card

C++

```
typedef struct tagEidPicture {
  BYTE picture[EID_MAX_PICTURE_LEN+1];
  int pictureLength;
} EidPicture, * PeidPicture;
```

File

CardStructures.h (see page 159)

Members

Members	Description
BYTE picture[EID_MAX_PICTURE_LEN+1];	Picture raw data buffer
int pictureLength;	Picture raw data buffer length

1.2.23 PSISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordA {
  char Name[SIS_MAX_NAME_LEN + 1];
  char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  char Initial[SIS_MAX_INITIAL_LEN+ 1];
  char Sex[SIS_MAX_SEX_LEN + 1];
  char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  char CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordA, * PSISRecordA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
char Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner

char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.24 PSISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordW {
  WCHAR Name[SIS_MAX_NAME_LEN + 1];
  WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
  WCHAR Sex[SIS_MAX_SEX_LEN + 1];
  WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  WCHAR CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordW, * PSISRecordW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.25 SISRecordA Structure

Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordA {
  char Name[SIS_MAX_NAME_LEN + 1];
  char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  char Initial[SIS_MAX_INITIAL_LEN+ 1];
  char Sex[SIS_MAX_SEX_LEN + 1];
  char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  char CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordA, * PSISRecordA;
```

File

CardStructures.h (see page 159)

Members

Members	Description
char Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
char FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
char Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
char Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
char BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
char SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
char CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
char ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
char ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
char CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
char CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.2.26 SISRecordW Structure

Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

C++

```
typedef struct tagSISRecordW {
  WCHAR Name[SIS_MAX_NAME_LEN + 1];
  WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];
  WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];
  WCHAR Sex[SIS_MAX_SEX_LEN + 1];
  WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];
  WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];
  WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];
  WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];
  WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];
  WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];
  WCHAR CardName[SIS_MAX_CARDNAME_LEN +1 ];
} SISRecordW, * PSISRecordW;
```

File

CardStructures.h (see page 159)

Members

Members	Description
WCHAR Name[SIS_MAX_NAME_LEN + 1];	Family name of the card owner
WCHAR FirstName[SIS_MAX_FIRSTNAMES_LEN + 1];	First name of the card owner
WCHAR Initial[SIS_MAX_INITIAL_LEN+ 1];	Initial of the card owner
WCHAR Sex[SIS_MAX_SEX_LEN + 1];	Sex of the card owner
WCHAR BirthDate[SIS_FIELD_MAX_BIRTHDATE_LEN + 1];	Birth date of the card owner
WCHAR SocialSecurityNumber[SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN + 1];	Social security number of the card owner
WCHAR CaptureDate[SIS_FIELD_MAX_CAPTUREDATE_LEN + 1];	Date of issue
WCHAR ValidityDateBegin[SIS_FIELD_MAX_VALIDBEGIN_LEN + 1];	Card validity begin date
WCHAR ValidityDateEnd[SIS_FIELD_MAX_VALIDEND_LEN + 1];	Card validity end date
WCHAR CardNumber[SIS_FIELD_MAX_CARDNUMBER_LEN + 1];	Card number
WCHAR CardName[SIS_MAX_CARDNAME_LEN +1];	Name of the card

1.3 Macros

The following table lists macros in this documentation.

Macros

Name	Description
EID_MAX_BIRTHDATE_LEN (see page 147)	Maximum length of the birthdate
EID_MAX_BIRTHPLACE_LEN (see page 148)	Maximum length of the birthplace
EID_MAX_BREXITMENTION1_LEN (see page 148)	Maximum length of the BREXIT mention field
EID_MAX_BREXITMENTION2_LEN (see page 148)	Maximum length of the BREXIT mention field
EID_MAX_CARD_NUMBER_LEN (see page 148)	Maximum length of the card number field
EID_MAX_CERT_LEN (see page 148)	Maximum length of the certificate data
EID_MAX_CHIP_NUMBER_LEN (see page 149)	Maximum length of the chip number field
EID_MAX_DATE_BEGIN_LEN (see page 149)	Maximum length of the begin date field
EID_MAX_DATE_END_LEN (see page 149)	Maximum length of the end date field
EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN (see page 149)	Maximum length of theDate and country of protection field
EID_MAX_DELIVERY_MUNICIPALITY_LEN (see page 150)	Maximum length of the name of the devivery municipality
EID_MAX_DOCUMENT_TYPE_LEN (see page 150)	Maximum length of the document type field
EID_MAX_DUPLICATE_LEN (see page 150)	Maximum length of the Duplicate field
EID_MAX_FIRST_NAME1_LEN (see page 150)	Maximum length of the first name
EID_MAX_FIRST_NAME2_LEN (see page 151)	Maximum length of the first name
EID_MAX_MEMBEROFFAMILY_LEN (see page 151)	Maximum length of the Member of family field
EID_MAX_MUNICIPALITY_LEN (see page 151)	Maximum length of the municipality name field
EID_MAX_NAME_LEN (see page 151)	Maximum length of the surname
EID_MAX_NATIONAL_NUMBER_LEN (see page 151)	Maximum length of the national number
EID_MAX_NATIONALITY_LEN (see page 152)	Maximum length of the nationality

EID_MAX_NOBLE_CONDITION_LEN (see page 152)	Maximum length of the noble condition field
EID_MAX_PICTURE_LEN (see page 152)	Maximum length of the picture data
EID_MAX_REGIONALFILENUMBER_LEN (see page 152)	Maximum length of regional file number field
EID_MAX_SEX_LEN (see page 153)	Maximum length of the sex field
EID_MAX_SPECIAL_STATUS_LEN (see page 153)	Maximum length of the special status field
EID_MAX_SPECIALORGANIZATION_LEN (see page 153)	Maximum length of the Special organization field
EID_MAX_STREET_LEN (see page 153)	Maximum length of the street name field
EID_MAX_VAT1_LEN (see page 153)	Maximum length of the VAT1 field
EID_MAX_VAT2_LEN (see page 154)	Maximum length of the VAT2 field
EID_MAX_WORKPERMITTYPE_LEN (see page 154)	Maximum length of the type of the workpermit
EID_MAX_ZIP_LEN (see page 154)	Maximum length of the ZIP code field
FONT_BOLD (see page 154)	This is macro FONT_BOLD.
FONT_ITALIC (see page 155)	This is macro FONT_ITALIC.
FONT_NORMAL (see page 155)	This is macro FONT_NORMAL.
FONT_STRIKEOUT (see page 155)	This is macro FONT_STRIKEOUT.
FONT_UNDERLINE (see page 155)	This is macro FONT_UNDERLINE.
IID_PPV_ARG (see page 155)	IID_PPV_ARG(IType, ppType) IType is the type of pType ppType is the variable of type IType that will be filled RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used. macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C
SIS_FIELD_MAX_BIRTHDATE_LEN (see page 156)	Maximum length of the birth date field
SIS_FIELD_MAX_CAPTUREDATE_LEN (see page 156)	Maximum length of the capture date field
SIS_FIELD_MAX_CARDNUMBER_LEN (see page 156)	Maximum length of the car number field
SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN (see page 156)	Maximum length of the social security number field
SIS_FIELD_MAX_VALIDBEGIN_LEN (see page 157)	Maximum length of the start validity date field
SIS_FIELD_MAX_VALIDEND_LEN (see page 157)	Maximum length of the end validity date field
SIS_MAX_CARDNAME_LEN (see page 157)	Maximum length of the card name field
SIS_MAX_FIRSTNAMES_LEN (see page 157)	Maximum length of the first name field
SIS_MAX_INITIAL_LEN (see page 158)	Maximum length of the initial field
SIS_MAX_NAME_LEN (see page 158)	Maximum length of the surname field
SIS_MAX_SEX_LEN (see page 158)	Maximum length of the sex field
WIDTHBYTES (see page 158)	This is macro WIDTHBYTES.

1.3.1 EID_MAX_BIRTHDATE_LEN Macro

Maximum length of the birthdate

C++

```
#define EID_MAX_BIRTHDATE_LEN 0xc
```

File

CardStructures.h (see page 159)

1.3.2 EID_MAX_BIRTHPLACE_LEN Macro

Maximum length of the birthplace

C++

```
#define EID_MAX_BIRTHPLACE_LEN 0x50
```

File

CardStructures.h (see page 159)

1.3.3 EID_MAX_BREXITMENTION1_LEN Macro

Maximum length of the BREXIT mention field

C++

```
#define EID_MAX_BREXITMENTION1_LEN 1
```

File

CardStructures.h (see page 159)

1.3.4 EID_MAX_BREXITMENTION2_LEN Macro

Maximum length of the BREXIT mention field

C++

```
#define EID_MAX_BREXITMENTION2_LEN 1
```

File

CardStructures.h (see page 159)

1.3.5 EID_MAX_CARD_NUMBER_LEN Macro

Maximum length of the card number field

C++

```
#define EID_MAX_CARD_NUMBER_LEN 0xc
```

File

CardStructures.h (see page 159)

1.3.6 EID_MAX_CERT_LEN Macro

Maximum length of the certificate data

C++

```
#define EID_MAX_CERT_LEN 0x800
```

File

CardStructures.h (see page 159)

1.3.7 EID_MAX_CHIP_NUMBER_LEN Macro

Maximum length of the chip number field

C++

```
#define EID_MAX_CHIP_NUMBER_LEN 0x20
```

File

CardStructures.h (see page 159)

1.3.8 EID_MAX_DATE_BEGIN_LEN Macro

Maximum length of the begin date field

C++

```
#define EID_MAX_DATE_BEGIN_LEN 0xa
```

File

CardStructures.h (see page 159)

1.3.9 EID_MAX_DATE_END_LEN Macro

Maximum length of the end date field

C++

```
#define EID_MAX_DATE_END_LEN 0xa
```

File

CardStructures.h (see page 159)

1.3.10 EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN Macro

Maximum length of theDate and country of protection field

C++

```
#define EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN 13
```

File

CardStructures.h (see page 159)

1.3.11 EID_MAX_DELIVERY_MUNICIPALITY_LEN Macro

Maximum length of the name of the devivery municipality

C++

```
#define EID_MAX_DELIVERY_MUNICIPALITY_LEN 0x50
```

File

CardStructures.h (see page 159)

1.3.12 EID_MAX_DOCUMENT_TYPE_LEN Macro

Maximum length of the document type field

C++

```
#define EID_MAX_DOCUMENT_TYPE_LEN 0x2
```

File

CardStructures.h (see page 159)

1.3.13 EID_MAX_DUPLICATE_LEN Macro

Maximum length of the Duplicate field

C++

```
#define EID_MAX_DUPLICATE_LEN 2
```

File

CardStructures.h (see page 159)

1.3.14 EID_MAX_FIRST_NAME1_LEN Macro

Maximum length of the first name

C++

```
#define EID_MAX_FIRST_NAME1_LEN 0x5f
```

File

CardStructures.h (see page 159)

1.3.15 EID_MAX_FIRST_NAME2_LEN Macro

Maximum length of the first name

C++

```
#define EID_MAX_FIRST_NAME2_LEN 0x3
```

File

CardStructures.h (see page 159)

1.3.16 EID_MAX_MEMBEROFFAMILY_LEN Macro

Maximum length of the Member of family field

C++

```
#define EID_MAX_MEMBEROFFAMILY_LEN 1
```

File

CardStructures.h (see page 159)

1.3.17 EID_MAX_MUNICIPALITY_LEN Macro

Maximum length of the municipality name field

C++

```
#define EID_MAX_MUNICIPALITY_LEN 0x43
```

File

CardStructures.h (see page 159)

1.3.18 EID_MAX_NAME_LEN Macro

Maximum length of the surname

C++

```
#define EID_MAX_NAME_LEN 0x6e
```

File

CardStructures.h (see page 159)

1.3.19 EID_MAX_NATIONAL_NUMBER_LEN Macro

Maximum length of the national number

C++

```
#define EID_MAX_NATIONAL_NUMBER_LEN 0xb
```

File

CardStructures.h (see page 159)

1.3.20 EID_MAX_NATIONALITY_LEN Macro

Maximum length of the nationality

C++

```
#define EID_MAX_NATIONALITY_LEN 0x55
```

File

CardStructures.h (see page 159)

1.3.21 EID_MAX_NOBLE_CONDITION_LEN Macro

Maximum length of the noble condition field

C++

```
#define EID_MAX_NOBLE_CONDITION_LEN 0x32
```

File

CardStructures.h (see page 159)

1.3.22 EID_MAX_PICTURE_LEN Macro

Maximum length of the picture data

C++

```
#define EID_MAX_PICTURE_LEN 0x1000
```

File

CardStructures.h (see page 159)

1.3.23 EID_MAX_REGIONALFILENUMBER_LEN Macro

Maximum length of regional file number field

C++

```
#define EID_MAX_REGIONALFILENUMBER_LEN 18
```

File

CardStructures.h (see page 159)

1.3.24 EID_MAX_SEX_LEN Macro

Maximum length of the sex field

C++

```
#define EID_MAX_SEX_LEN 0x1
```

File

CardStructures.h (see page 159)

1.3.25 EID_MAX_SPECIAL_STATUS_LEN Macro

Maximum length of the special status field

C++

```
#define EID_MAX_SPECIAL_STATUS_LEN 0x2
```

File

CardStructures.h (see page 159)

1.3.26 EID_MAX_SPECIALORGANIZATION_LEN Macro

Maximum length of the Special organization field

C++

```
#define EID_MAX_SPECIALORGANIZATION_LEN 1
```

File

CardStructures.h (see page 159)

1.3.27 EID_MAX_STREET_LEN Macro

Maximum length of the street name field

C++

```
#define EID_MAX_STREET_LEN 0x50
```

File

CardStructures.h (see page 159)

1.3.28 EID_MAX_VAT1_LEN Macro

Maximum length of the VAT1 field

C++

```
#define EID_MAX_VAT1_LEN 13
```

File

CardStructures.h (see page 159)

1.3.29 EID_MAX_VAT2_LEN Macro

Maximum length of the VAT2 field

C++

```
#define EID_MAX_VAT2_LEN 13
```

File

CardStructures.h (see page 159)

1.3.30 EID_MAX_WORKPERMITTYPE_LEN Macro

Maximum length of the type of the workpermit

C++

```
#define EID_MAX_WORKPERMITTYPE_LEN 1
```

File

CardStructures.h (see page 159)

1.3.31 EID_MAX_ZIP_LEN Macro

Maximum length of the ZIP code field

C++

```
#define EID_MAX_ZIP_LEN 0x4
```

File

CardStructures.h (see page 159)

1.3.32 FONT_BOLD Macro

This is macro FONT_BOLD.

C++

```
#define FONT_BOLD 0x01
```

File

Graphics.h (see page 164)

1.3.33 FONT_ITALIC Macro

This is macro FONT_ITALIC.

C++

```
#define FONT_ITALIC 0x02
```

File

Graphics.h (see page 164)

1.3.34 FONT_NORMAL Macro

This is macro FONT_NORMAL.

C++

```
#define FONT_NORMAL 0x00
```

File

Graphics.h (see page 164)

1.3.35 FONT_STRIKEOUT Macro

This is macro FONT_STRIKEOUT.

C++

```
#define FONT_STRIKEOUT 0x08
```

File

Graphics.h (see page 164)

1.3.36 FONT_UNDERLINE Macro

This is macro FONT_UNDERLINE.

C++

```
#define FONT_UNDERLINE 0x04
```

File

Graphics.h (see page 164)

1.3.37 IID_PPV_ARG Macro

IID_PPV_ARG(IType, ppType) IType is the type of pType ppType is the variable of type IType that will be filled

RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used.

macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C

C++

```
#define IID_PPV_ARG(IType, ppType) &IID_##IType, (void**)(ppType))
```

File

FileOperations.h (see page 162)

1.3.38 SIS_FIELD_MAX_BIRTHDATE_LEN Macro

Maximum length of the birth date field

C++

```
#define SIS_FIELD_MAX_BIRTHDATE_LEN 0x8
```

File

CardStructures.h (see page 159)

1.3.39 SIS_FIELD_MAX_CAPTUREDATE_LEN Macro

Maximum length of the capture date field

C++

```
#define SIS_FIELD_MAX_CAPTUREDATE_LEN 0x8
```

File

CardStructures.h (see page 159)

1.3.40 SIS_FIELD_MAX_CARDNUMBER_LEN Macro

Maximum length of the car number field

C++

```
#define SIS_FIELD_MAX_CARDNUMBER_LEN 0xa
```

File

CardStructures.h (see page 159)

1.3.41 SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN Macro

Maximum length of the social security number field

C++

```
#define SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN 0xb
```

File

CardStructures.h (see page 159)

1.3.42 SIS_FIELD_MAX_VALIDBEGIN_LEN Macro

Maximum length of the start validity date field

C++

```
#define SIS_FIELD_MAX_VALIDBEGIN_LEN 0x8
```

File

CardStructures.h (see page 159)

1.3.43 SIS_FIELD_MAX_VALIDEND_LEN Macro

Maximum length of the end validity date field

C++

```
#define SIS_FIELD_MAX_VALIDEND_LEN 0x8
```

File

CardStructures.h (see page 159)

1.3.44 SIS_MAX_CARDNAME_LEN Macro

Maximum length of the card name field

C++

```
#define SIS_MAX_CARDNAME_LEN 0x6
```

File

CardStructures.h (see page 159)

1.3.45 SIS_MAX_FIRSTNAMES_LEN Macro

Maximum length of the first name field

C++

```
#define SIS_MAX_FIRSTNAMES_LEN 0x18
```

File

CardStructures.h (see page 159)

1.3.46 SIS_MAX_INITIAL_LEN Macro

Maximum length of the initial field

C++

```
#define SIS_MAX_INITIAL_LEN 0x1
```

File

CardStructures.h (see page 159)

1.3.47 SIS_MAX_NAME_LEN Macro

Maximum length of the surname field

C++

```
#define SIS_MAX_NAME_LEN 0x30
```

File

CardStructures.h (see page 159)

1.3.48 SIS_MAX_SEX_LEN Macro

Maximum length of the sex field

C++

```
#define SIS_MAX_SEX_LEN 0x1
```

File

CardStructures.h (see page 159)

1.3.49 WIDTHBYTES Macro

This is macro WIDTHBYTES.

C++

```
#define WIDTHBYTES(i) (((i) + 31) / 32 * 4)
```

File

Graphics.h (see page 164)

1.4 Files

The following table lists files in this documentation.

Files

Name	Description
CardEvents.h (see page 159)	The definition of the possible card events
CardStructures.h (see page 159)	The definition of the information storage structures
Encryption.h (see page 161)	Encryption and decryption operations
FileOperations.h (see page 162)	Files and folders manipulations
Graphics.h (see page 164)	This is file Graphics.h.
NationalityConverter.h (see page 165)	Nationality to ISO code cenverter
quricol.h (see page 165)	QR Code generator
Swelio.h (see page 165)	Belgian electronic Id card access engine
System.h (see page 170)	Windows related routines
SystemInfo.h (see page 171)	Routines for querying information about operating system

1.4.1 CardEvents.h

The definition of the possible card events

Enumerations

	Name	Description
	tagCardEventType (see page 128)	The type of the reader event
	CardEventType (see page 134)	The type of the reader event

1.4.2 CardStructures.h

The definition of the information storage structures

Macros

Name	Description
EID_MAX_BIRTHDATE_LEN (see page 147)	Maximum length of the birthdate
EID_MAX_BIRTHPLACE_LEN (see page 148)	Maximum length of the birthplace
EID_MAX_BREXITMENTION1_LEN (see page 148)	Maximum length of the BREXIT mention field
EID_MAX_BREXITMENTION2_LEN (see page 148)	Maximum length of the BREXIT mention field
EID_MAX_CARD_NUMBER_LEN (see page 148)	Maximum length of the card number field
EID_MAX_CERT_LEN (see page 148)	Maximum length of the certificate data
EID_MAX_CHIP_NUMBER_LEN (see page 149)	Maximum length of the chip number field
EID_MAX_DATE_BEGIN_LEN (see page 149)	Maximum length of the begin date field
EID_MAX_DATE_END_LEN (see page 149)	Maximum length of the end date field
EID_MAX_DATEANDCOUNTRYOFPROTECTION_LEN (see page 149)	Maximum length of theDate and country of protection field
EID_MAX_DELIVERY_MUNICIPALITY_LEN (see page 150)	Maximum length of the name of the devivery municipality
EID_MAX_DOCUMENT_TYPE_LEN (see page 150)	Maximum length of the document type field

EID_MAX_DUPLICATE_LEN (see page 150)	Maximum length of the Duplicate field
EID_MAX_FIRST_NAME1_LEN (see page 150)	Maximum length of the first name
EID_MAX_FIRST_NAME2_LEN (see page 151)	Maximum length of the first name
EID_MAX_MEMBEROFFAMILY_LEN (see page 151)	Maximum length of the Member of family field
EID_MAX_MUNICIPALITY_LEN (see page 151)	Maximum length of the municipality name field
EID_MAX_NAME_LEN (see page 151)	Maximum length of the surname
EID_MAX_NATIONAL_NUMBER_LEN (see page 151)	Maximum length of the national number
EID_MAX_NATIONALITY_LEN (see page 152)	Maximum length of the nationality
EID_MAX_NOBLE_CONDITION_LEN (see page 152)	Maximum length of the noble condition field
EID_MAX_PICTURE_LEN (see page 152)	Maximum length of the picture data
EID_MAX_REGIONALFILENUMBER_LEN (see page 152)	Maximum length of regional file number field
EID_MAX_SEX_LEN (see page 153)	Maximum length of the sex field
EID_MAX_SPECIAL_STATUS_LEN (see page 153)	Maximum length of the special status field
EID_MAX_SPECIALORGANIZATION_LEN (see page 153)	Maximum length of the Special organization field
EID_MAX_STREET_LEN (see page 153)	Maximum length of the street name field
EID_MAX_VAT1_LEN (see page 153)	Maximum length of the VAT1 field
EID_MAX_VAT2_LEN (see page 154)	Maximum length of the VAT2 field
EID_MAX_WORKPERMITTYPE_LEN (see page 154)	Maximum length of the type of the workpermit
EID_MAX_ZIP_LEN (see page 154)	Maximum length of the ZIP code field
SIS_FIELD_MAX_BIRTHDATE_LEN (see page 156)	Maximum length of the birth date field
SIS_FIELD_MAX_CAPTUREDATE_LEN (see page 156)	Maximum length of the capture date field
SIS_FIELD_MAX_CARDNUMBER_LEN (see page 156)	Maximum length of the car number field
SIS_FIELD_MAX_SOCIAL_SECURITY_NUMBER_LEN (see page 156)	Maximum length of the social security number field
SIS_FIELD_MAX_VALIDBEGIN_LEN (see page 157)	Maximum length of the start validity date field
SIS_FIELD_MAX_VALIDEND_LEN (see page 157)	Maximum length of the end validity date field
SIS_MAX_CARDNAME_LEN (see page 157)	Maximum length of the card name field
SIS_MAX_FIRSTNAMES_LEN (see page 157)	Maximum length of the first name field
SIS_MAX_INITIAL_LEN (see page 158)	Maximum length of the initial field
SIS_MAX_NAME_LEN (see page 158)	Maximum length of the surname field
SIS_MAX_SEX_LEN (see page 158)	Maximum length of the sex field

Structures

	Name	Description
	tagEidAddressA (see page 128)	EID address information, stored on the card - ANSI version
	tagEidAddressW (see page 128)	EID address information, stored on the card - UNICODE version
	tagEidCertificate (see page 129)	Certificate, stored on EID card
	tagEidIdentityA (see page 129)	Identity information stored on EID card - ANSI version
	tagEidIdentityW (see page 131)	Identity information stored on EID card - UNICODE version
	tagEidPicture (see page 132)	Raw picture data from EID card
	tagSISRecordA (see page 132)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	tagSISRecordW (see page 133)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	EidAddressA (see page 134)	EID address information, stored on the card - ANSI version

	EidAddressW (see page 135)	EID address information, stored on the card - UNICODE version
	EidCertificate (see page 135)	Certificate, stored on EID card
	EidIdentityA (see page 135)	Identity information stored on EID card - ANSI version
	EidIdentityW (see page 137)	Identity information stored on EID card - UNICODE version
	EidPicture (see page 138)	Raw picture data from EID card
	PEidAddressA (see page 139)	EID address information, stored on the card - ANSI version
	PEidAddressW (see page 139)	EID address information, stored on the card - UNICODE version
	PEidCertificate (see page 139)	Certificate, stored on EID card
	PEidIdentityA (see page 140)	Identity information stored on EID card - ANSI version
	PEidIdentityW (see page 141)	Identity information stored on EID card - UNICODE version
	PeidPicture (see page 143)	Raw picture data from EID card
	PSISRecordA (see page 143)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	PSISRecordW (see page 144)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordA (see page 144)	Public information stored on Belgian SIS card - ANSI version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)
	SISRecordW (see page 145)	Public information stored on Belgian SIS card - UNICODE version. The SIS card is the social security card of each Belgian resident (Belgian or foreigner)

1.4.3 Encryption.h

Encryption and decryption operations

Functions

	Name	Description
	CardDecryptFileA (see page 15)	Decrypt file using Belgian Id card
	CardDecryptFileW (see page 15)	Decrypt file using Belgian Id card
	CardEncryptFileA (see page 16)	Encrypt file using Belgian Id card
	CardEncryptFileW (see page 16)	Encrypt file using Belgian Id card
	CheckMD5 (see page 19)	Checks the MD5 hash value of the memory buffer
	CheckSHA1 (see page 19)	Checks the SHA1 hash value of the memory buffer
	CheckSHA256 (see page 20)	Checks the SHA256 hash value of the memory buffer
	DecryptFileAESA (see page 26)	Decrypts file using AES algorithm.
	DecryptFileAESW (see page 27)	Decrypts file using AES algorithm.
	EncryptFileAESA (see page 34)	Encrypts file using AES algorithm.
	EncryptFileAESW (see page 35)	Encrypts file using AES algorithm.
	GetFileMD5A (see page 58)	Gets the MD5 hash value for the file
	GetFileMD5W (see page 58)	Gets the MD5 hash value for the file
	GetFileSHA1A (see page 60)	Gets the SHA1 hash value for the file
	GetFileSHA1W (see page 60)	Gets the SHA1 hash value for the file
	GetFileSHA256A (see page 60)	Gets the SHA256 hash value for the file
	GetFileSHA256W (see page 61)	Gets the SHA256 hash value for the file
	GetMD5 (see page 63)	Gets the MD5 hash value for the content of the memory buffer
	GetSHA1 (see page 68)	Gets the SHA1 hash value for the content of the memory buffer
	GetSHA256 (see page 68)	Gets the SHA256 hash value for the content of the memory buffer

1.4.4 FileOperations.h

Files and folders manipulations

Functions

	Name	Description
	AllocateBuffer (see page 11)	Allocates the buffer in memory
	ClearFileAttributesA (see page 20)	This function sets the file attributes to normal.
	ClearFileAttributesW (see page 20)	This function sets the file attributes to normal.
	CreateUnicodeFileA (see page 23)	Creates UNICODE file
	CreateUnicodeFileW (see page 23)	Creates UNICODE file
	DeallocateBuffer (see page 25)	Deallocates the memory buffer
	DeleteToRecycleBinA (see page 27)	Deletes file to the Windows Recycle Bin
	DeleteToRecycleBinW (see page 28)	Deletes file to WIndows Recycle Bin
	DirectoryExistsA (see page 29)	Determines whether a specified directory exists.
	DirectoryExistsW (see page 29)	Determines whether a specified directory exists.
	FileCloseA (see page 35)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
	FileCloseW (see page 35)	Concludes input/output (I/O) to a file opened using the FileCreateRewrite function.
	FileCopyA (see page 36)	The CopyFile function copies an existing file to a new file.
	FileCopyW (see page 36)	The CopyFile function copies an existing file to a new file.
	FileCreateRewriteA (see page 37)	Creates new or overwrites existing file
	FileCreateRewriteW (see page 37)	Creates new or overwrites existing file
	FileDeleteA (see page 37)	Deletes a file from disk.
	FileDeleteW (see page 38)	Deletes a file from disk.
	FileExistsA (see page 38)	Tests whether a specified file exists.
	FileExistsW (see page 38)	Tests whether a specified file exists.
	FileExtensionIsA (see page 39)	Checks the file extension
	FileExtensionIsW (see page 39)	Checks the file extension
	FileGetSizeA (see page 39)	Retrieves the size of a specified file.
	FileGetSizeW (see page 40)	Retrieves the size of a specified file.
	FileIsExeA (see page 40)	Checks if the file is a Windows executable
	FileIsExeW (see page 41)	Checks if the file is a Windows executable
	FileIsIconA (see page 41)	Checks if the file is a Windows icon (.ico) file
	FileIsIconW (see page 41)	Checks if the file is a Windows icon (.ico) file
	FileIsImageA (see page 42)	Checks if the file is an image file
	FileIsImageW (see page 42)	Checks if the file is an image file
	FileIsLink (see page 42)	Checks to see if the file specified by file name is a Microsoft Windows shortcut (.Lnk) file (and is neither a file nor a folder).
	FileOrFolderExistsA (see page 43)	Checks if the file or folder with the given name exists

	FileOrFolderExistsW (see page 43)	Checks if the file or folder with the given name exists
	FileRenameA (see page 43)	Renames the file
	FileRenameW (see page 44)	Renames the file
	FileWriteA (see page 44)	Writes string to the file
	FileWriteCharA (see page 44)	Writes one character to the file
	FileWriteCharW (see page 45)	Writes one character to the file
	FileWriteNewLineA (see page 45)	Writes new line sequence to the file
	FileWriteNewLineW (see page 45)	Writes new line sequence to the file
	FileWriteW (see page 46)	Writes string to the file
	FullPathA (see page 47)	Gets the full path to the file based on file name
	FullPathW (see page 47)	Gets the full path to the file based on file name
	GetAllFiles (see page 55)	Returns the names of files in a specified directory.
	GetFilesCountA (see page 59)	Calculates the number of files in the given folder
	GetFilesCountW (see page 59)	Calculates the number of files in the given folder
	IsAnimatedGIFA (see page 71)	Checks if the file is an animated GIF image file
	IsAnimatedGIFW (see page 71)	Checks if the file is an animated GIF image file
	IsDirectoryA (see page 74)	Verifies that a path is a valid directory.
	IsDirectoryW (see page 74)	Verifies that a path is a valid directory.
	IsUnicodeFileA (see page 79)	Checks if the file is UNICODE file
	IsUnicodeFileW (see page 80)	Checks if the file is UNICODE file
	IsValidFileNameA (see page 80)	Checks if provided string is a valid file name
	IsValidFileNameW (see page 80)	Checks if provided string is a valid file name
	IsValidPathNameA (see page 81)	Checks if provided string is a valid file path
	IsValidPathNameW (see page 81)	Checks if provided string is a valid file path
	ReadBufferFromFileA (see page 91)	Reads the content of the file to the memory buffer
	ReadBufferFromFileW (see page 91)	Reads the content of the file to the memory buffer
	ShellCopyFileA (see page 120)	Copies file to the new location
	ShellCopyFileW (see page 120)	Copies file to the new location
	StripFileNameA (see page 122)	Replaces environment variable names with values
	StripFileNameW (see page 123)	Replaces environment variable names with values
	WriteBufferToFileA (see page 126)	Writes the memory buffer to file
	WriteBufferToFileW (see page 126)	Writes the memory buffer to file

Macros

Name	Description
IID_PPV_ARG (see page 155)	IID_PPV_ARG(IType, ppType) IType is the type of pType ppType is the variable of type IType that will be filled RESULTS in: IID_IType, ppvType will create a compiler error if wrong level of indirection is used. macro for QueryInterface and related functions that require a IID and a (void **) this will insure that the cast is safe and appropriate on C

1.4.5 Graphics.h

This is file Graphics.h.

Functions

	Name	Description
	AlphaBlendBitmap (see page 14)	This is function AlphaBlendBitmap.
	AlphaBlendNative (see page 15)	This is function AlphaBlendNative.
	CloneFont (see page 21)	This is function CloneFont.
	CopyNativeBitmap (see page 22)	This is function CopyNativeBitmap.
	CreateNativeBitmap (see page 23)	This is function CreateNativeBitmap.
	CreateWindowsFont (see page 24)	This is function CreateWindowsFont.
	DestroyFont (see page 28)	This is function DestroyFont.
	DpiY (see page 30)	This is function DpiY.
	DrawAlphaText (see page 30)	This is function DrawAlphaText.
	DrawAlphaTextRect (see page 30)	This is function DrawAlphaTextRect.
	DrawNativeBitmap (see page 31)	This is function DrawNativeBitmap.
	DrawTextDirect (see page 31)	This is function DrawTextDirect.
	DrawTextDirectEx (see page 32)	This is function DrawTextDirectEx.
	DrawTextGlow (see page 32)	This is function DrawTextGlow.
	DrawTextLine (see page 32)	This is function DrawTextLine.
	DrawTextOutline (see page 32)	This is function DrawTextOutline.
	DrawTextRect (see page 33)	This is function DrawTextRect.
	EmToPixels (see page 33)	This is function EmToPixels.
	GetTextLineSize (see page 69)	This is function GetTextLineSize.
	GetTextSize (see page 70)	This is function GetTextSize.
	GetTextSizeEx (see page 70)	This is function GetTextSizeEx.
	LoadBitmapJPG (see page 84)	This is function LoadBitmapJPG.
	LoadBitmapPNG (see page 84)	This is function LoadBitmapPNG.
	LoadPNGResource (see page 86)	This is function LoadPNGResource.
	MakeCompatibleBitmap (see page 87)	This is function MakeCompatibleBitmap.
	PointsToPixels (see page 88)	This is function PointsToPixels.
	StretchNativeBitmap (see page 122)	This is function StretchNativeBitmap.

Macros

Name	Description
FONT_BOLD (see page 154)	This is macro FONT_BOLD.
FONT_ITALIC (see page 155)	This is macro FONT_ITALIC.
FONT_NORMAL (see page 155)	This is macro FONT_NORMAL.
FONT_STRIKEOUT (see page 155)	This is macro FONT_STRIKEOUT.
FONT_UNDERLINE (see page 155)	This is macro FONT_UNDERLINE.
WIDTHBYTES (see page 158)	This is macro WIDTHBYTES.

1.4.6 NationalityConverter.h

Nationality to ISO code cenverter

Functions

	Name	Description
	GetISOCodeA (see page 62)	Returns the country ISO code based on the nationality string
	GetISOCodeW (see page 63)	Returns the country ISO code based on the nationality string

1.4.7 quricol.h

QR Code generator

Functions

	Name	Description
	DestroyImageBuffer (see page 28)	Destroys the memory buffer
	GenerateBMPA (see page 49)	Generates Windows Bitmap file with QR Code image
	GenerateBMPW (see page 50)	Generates Windows Bitmap file with QR Code image
	GeneratePNGA (see page 52)	Generates PNG file with QR Code image
	GeneratePNGW (see page 53)	Generates PNG file with QR Code image
	GetHBitmapA (see page 61)	Generates Windows Bitmap in memory with QR Code image
	GetHBitmapW (see page 62)	Generates Windows Bitmap in memory with QR Code image
	GetPNGA (see page 64)	Writes PNG image to the memory buffer.
	GetPNGW (see page 64)	Writes PNG image to the memory buffer.

1.4.8 Swelio.h

Belgian electronic Id card access engine

Functions

	Name	Description
	ActivateCard (see page 9)	Established communication between the card and the reader
	ActivateCardEx (see page 10)	Established communication between the card and the reader
	AddFileToContainer (see page 10)	Add existing file to the container
	CardSignCadesT (see page 17)	Sign data with eID card according to CADES-T standard
	CardSignCMS (see page 17)	Sign data with eID card according to CMS standard
	CertSignCadesT (see page 18)	Sign data with the certificate file according to CADES-T standard
	CertSignCMS (see page 18)	Sign data with the certificate file according to CMS standard
	ContainerCertificate (see page 21)	Assign certificate for signing ASIC container
	ContainerEidCertificate (see page 22)	Select EID card certificate to sign ASIC container
	ContainerPickCertificate (see page 22)	Pick certificate to sign ASIC container

	CreateCardBuffer (see page 23)	Creates XML buffer
	DeactivateCard (see page 24)	Terminates a cennection between a smart card and a reader
	DeactivateCardEx (see page 25)	Terminates a connection between a smart card and a reader
	DeleteCardBuffer (see page 27)	Deletes XML buffer
	DisplayCertificate (see page 30)	Displays the dialog window with certificate information
	EncodeCertificate (see page 33)	Performs Base64 encoding of the certificate
	EncodePhoto (see page 34)	Performs Base64 encoding of the photo
	FreeContainer (see page 46)	Deallocates ASIC container
	GenerateAuthenticationSignatureA (see page 47)	Generate authentication signature
	GenerateAuthenticationSignatureExA (see page 48)	Generate authentication signature
	GenerateAuthenticationSignatureExW (see page 48)	Generate authentication signature
	GenerateAuthenticationSignatureW (see page 49)	Generate authentication signature
	GenerateNonRepudiationSignatureA (see page 50)	Generate non repudiation signature
	GenerateNonRepudiationSignatureExA (see page 51)	Generate non repudiation signature
	GenerateNonRepudiationSignatureExW (see page 51)	Generate non repudiation signature
	GenerateNonRepudiationSignatureW (see page 52)	Generate non repudiation signature
	GenerateQRCodeA (see page 53)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeExA (see page 53)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeExW (see page 54)	Read eID card and save the identity information and address to PNG QR Code file
	GenerateQRCodeW (see page 54)	Read eID card and save the identity information and address to PNG QR Code file
	GetCardBufferA (see page 55)	Gets XML or CSV information from the memory buffer
	GetCardBufferSize (see page 56)	This function returns the size of the buffer needed to hold the information from the eID card in the XML or CSV format
	GetCardBufferW (see page 56)	Gets XML or CSV information from the memory buffer
	GetCardSerialNumber (see page 56)	Get the serial number of EID card
	GetCardVersion (see page 57)	Get the applet version number for card in the reader with specified number
	GetEncodedCertificateSize (see page 57)	Returns the size of the Base64 encoded certificate
	GetEncodedPhotoSize (see page 58)	Calculates buffer size for Base64 encoded photo
	GetReaderIndexA (see page 65)	Returns the zero-based reader index with specified name
	GetReaderIndexW (see page 65)	Returns the zero-based reader index with specified name
	GetReaderNameA (see page 65)	Returns the name of the card reader
	GetReaderNameLenA (see page 66)	Returns the length of the reader name
	GetReaderNameLenW (see page 66)	Returns the length of the reader name
	GetReaderNameW (see page 66)	Returns the name of the card reader
	GetReadersCount (see page 67)	Get number of card readers connected to PC
	GetSelectedReaderIndex (see page 67)	Returns the index of the active smart card reader
	GetSupportSIS (see page 69)	Checks if the SIS cards are supported by the engine
	InitializeContainer (see page 70)	Initializes ASIC container

	IsCardActivated (see page 71)	Checks the connection between a smart card and a reader
	IsCardActivatedEx (see page 72)	Checks the connection between a smart card and a reader
	IsCardPresent (see page 72)	Checks if the card is present in the card reader
	IsCardPresentEx (see page 72)	Checks if the card is present in the card reader
	IsCardStillInserted (see page 73)	Checks if the card is still inserted in the card reader
	IsCardStillInsertedEx (see page 73)	Checks if the card is still inserted in the card reader
	IsEIDCard (see page 74)	Check if Belgian EID card is inserted into card reader
	IsEIDCardEx (see page 75)	Check if Belgian EID card is inserted into card reader
	IsEngineActive (see page 75)	Checks if the Swelio Engine is activated
	IsFemaleA (see page 76)	Checks if the card owner is female
	IsFemaleW (see page 76)	Checks if the card owner is female
	IsMaleA (see page 76)	Checks if the card owner is male
	IsMaleW (see page 77)	Checks if the card owner is male
	IsSISCard (see page 78)	Check if Belgian SIS card is inserted into card reader
	IsSISCardEx (see page 79)	Check if Belgian SIS card is inserted into card reader
	LoadCertificateA (see page 84)	Reads the certificate from a file
	LoadCertificateW (see page 84)	Reads the certificate from a file
	LoadIdentityA (see page 85)	Reads the raw identity information from a file
	LoadIdentityW (see page 85)	Reads the raw identity information from a file
	LoadPhotoA (see page 86)	Loads photo from a file
	LoadPhotoW (see page 86)	Loads photo from a file
	ReadAddressA (see page 89)	Read address information from Belgian eID card
	ReadAddressExA (see page 89)	Read address information from Belgian eID card
	ReadAddressExW (see page 89)	Read address information from Belgian eID card
	ReadAddressW (see page 90)	Read address information from Belgian eID card
	ReadAuthenticationCertificate (see page 90)	Read Authentication Certificate to memory
	ReadAuthenticationCertificateEx (see page 90)	Read Authentication Certificate to memory
	ReadCaCertificate (see page 92)	Read Ca Certificate to memory
	ReadCaCertificateEx (see page 92)	Read Ca Certificate to memory
	ReadIdentityA (see page 92)	Read identity information from Belgian eID card
	ReadIdentityExA (see page 93)	Read identity information from Belgian eID card
	ReadIdentityExW (see page 93)	Read identity information from Belgian eID card
	ReadIdentityW (see page 94)	Read identity information from Belgian eID card
	ReadNonRepudiationCertificate (see page 94)	Read Non Repudiation Certificate to memory
	ReadNonRepudiationCertificateEx (see page 94)	Read Non Repudiation Certificate to memory
	ReadPhoto (see page 95)	Reads a photo from a card
	ReadPhotoAsBitmap (see page 95)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
	ReadPhotoAsBitmapEx (see page 96)	Reads the picture from the card, converts it to bitmap and returns the bitmap handle Decription: Reads the photo from the Belgian eID card and returns the Windows bitmap handle Reading the photo from the card is a time consuming operation. Do it only when needed.
	ReadPhotoEx (see page 96)	Reads a photo from a card
	ReadRootCaCertificate (see page 96)	Read Root Ca Certificate to memory
	ReadRootCaCertificateEx (see page 97)	Read Root Ca Certificate to memory

	ReadRrnCertificate (see page 97)	Read Rrn Certificate to memory
	ReadRrnCertificateEx (see page 98)	Read Rrn Certificate to memory
	ReadSISCardA (see page 98)	Read Belgian SIS card.
	ReadSISCardExA (see page 98)	Read Belgian SIS card.
	ReadSISCardExW (see page 99)	Read Belgian SIS card.
	ReadSISCardW (see page 99)	Read Belgian SIS card.
	ReloadReadersList (see page 100)	Reloads the list of the available card readers
	RemoveCallback (see page 100)	Remove callback procedure for card events
	SaveAuthenticationCertificateA (see page 102)	Save Authentication Certificate to a file
	SaveAuthenticationCertificateExW (see page 102)	Save Authentication Certificate to a file
	SaveAuthenticationCertificateW (see page 102)	Save Authentication Certificate to a file
	SaveCaCertificateA (see page 103)	Save Ca Certificate to a file
	SaveCaCertificateExW (see page 103)	Save Ca Certificate to a file
	SaveCaCertificateW (see page 103)	Save Ca Certificate to a file
	SaveCardToToXMLStreamExA (see page 104)	Read eID card and save the information to XML buffer
	SaveCardToToXMLStreamExW (see page 104)	Read eID card and save the information to XML buffer
	SaveCardToXmlA (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlExA (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlExW (see page 105)	Read eID card and save the information to XML file
	SaveCardToXmlW (see page 106)	Read eID card and save the information to XML file
	SaveContainer (see page 106)	Save container to the file
	SaveIdentityA (see page 107)	Saves indentity infornation to a file
	SaveIdentityW (see page 107)	Saves indentity infornation to a file
	SaveNonRepudiationCertificateA (see page 107)	Save Non Repudiation Certificate to a file
	SaveNonRepudiationCertificateExW (see page 108)	Save Non Repudiation Certificate to a file
	SaveNonRepudiationCertificateW (see page 108)	Save Non Repudiation Certificate to a file
	SavePersonCsvToStreamA (see page 108)	Read eID card and save the identity information to CSV memory buffer
	SavePersonCsvToStreamW (see page 109)	Read eID card and save the identity information to CSV memory buffer
	SavePersonToCsvA (see page 109)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvExA (see page 110)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvExW (see page 110)	Read eID card and save the identity information and address to CSV file
	SavePersonToCsvW (see page 111)	Read eID card and save the identity information and address to CSV file
	SavePhotoA (see page 111)	Save photo to a file
	SavePhotoAsBitmapA (see page 111)	Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.

	SavePhotoAsBitmapExA (see page 112)	Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsBitmapExW (see page 112)	Reads the picture from the card and saves it to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsBitmapW (see page 113)	Save the picture from the card to Windows Bitmap file Decription: Reads the photo from the Belgian eID card and writes it to the file as bitmap image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegA (see page 113)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegExA (see page 113)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegExW (see page 114)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoAsJpegW (see page 114)	Save the picture from the card to JPG file Decription: Reads the photo from the Belgian eID card and writes it to the file as JPG image. Reading the photo from the card is a time consuming operation. Do it only when needed.
	SavePhotoW (see page 114)	Saves photo to a file
	SaveRootCaCertificateA (see page 115)	Save Root Ca Certificate to a file
	SaveRootCaCertificateExW (see page 115)	Save Root Ca Certificate to a file
	SaveRootCaCertificateW (see page 116)	Save Root Ca Certificate to a file
	SaveRrnCertificateA (see page 116)	Save RRN Certificate to a file
	SaveRrnCertificateExW (see page 116)	Save RRN Certificate to a file
	SaveRrnCertificateW (see page 117)	Save RRN Certificate to a file
	SelectReader (see page 117)	When more than 1 reader connected, select the reader with specified number
	SelectReaderByNameA (see page 117)	Select active smart card reader by providing the reader name
	SelectReaderByNameW (see page 118)	Select active smart card reader by providing the reader name
	SendAPDU (see page 118)	This is function SendAPDU.
	SetCallback (see page 118)	Activates callback procedure for card status change event
	SetMWCompatibility (see page 119)	Set the compatibility mode with the old version of the oficial EID MiddleWare
	SetSupportSIS (see page 120)	Activates or deactivates SIS card support by engine
	StartEngine (see page 121)	Activates the Swelio Engine.
	StopEngine (see page 122)	Deactivates the Swelio Engine
	VerifyPinA (see page 124)	Verify PIN code
	VerifyPinExA (see page 124)	Verify PIN code
	VerifyPinExW (see page 125)	Verify PIN code
	VerifyPinW (see page 125)	Verify PIN code

	VerifySignature (see page 125)	Verifies the signature from the specified hash value.

1.4.9 System.h

Windows related routines

Functions

	Name	Description
	AddRemoveMessageFilter (see page 11)	Adds or removes a message from the User Interface Privilege Isolation (UIPI) message filter.
	AllocateDefaultHWNDA (see page 12)	This function creates the invisible tool window
	AllocateDefaultHWNDW (see page 12)	This function creates the invisible tool window
	AllocateHWNDA (see page 12)	This function creates the invisible tool window using the provided window procedure
	AllocateHWNDW (see page 13)	This function creates the invisible tool window using the provided window procedure
	AllocateLayeredWindowA (see page 13)	This function creates the layered window using the provided window class name
	AllocateLayeredWindowW (see page 13)	This function creates the layered window using the provided window class name
	AllocateWindowClassA (see page 14)	This function creates the standard window using the provided window class name
	AllocateWindowClassW (see page 14)	This function creates the standard window using the provided window class name
	BringWindowToFront (see page 15)	This function brings the specified window to the top of the z-order.
	ClearUnusedMemory (see page 21)	Clears unused memory and minimized the application memory usage
	DeallocateHWNDA (see page 25)	This function destroys the specified window.
	DeallocateHWNDW (see page 26)	This function destroys the specified window.
	DrawLayeredWindow (see page 31)	Repaints the surface of the layered window
	EmptyRecycleBin (see page 33)	Empties the recycle bin
	fpreset (see page 46)	This is function fpreset.
	GetStartupA (see page 68)	Checks if the application is registered to run when Windows starts
	GetStartupW (see page 69)	Checks if the application is registered to run when Windows starts
	HibernateWindows (see page 70)	Hibernates Windows
	LayeredWndProcA (see page 83)	The default window procedure for the layered window
	LayeredWndProcW (see page 83)	The default window procedure for the layered window
	MakeSoundFromFileA (see page 87)	Plays the wave sound from the file
	MakeSoundFromFileW (see page 87)	Plays the wave sound from the file
	MakeSoundFromResourceA (see page 87)	Plays the wave sound from the resource
	MakeSoundFromResourceW (see page 88)	Plays the wave sound from the resource
	RemoveStartupA (see page 100)	Removes the application from the list of the automatically started applications

	RemoveStartupW (see page 101)	Removes the application from the list of the automatically started applications
	RestoreWindowSubclassA (see page 101)	Restores window standard procedure
	RestoreWindowSubclassW (see page 101)	Restores window standard procedure
	SetStartupA (see page 119)	Register application to run when Windows starts
	SetStartupW (see page 119)	Register application to run when Windows starts
	ShutdownWindows (see page 121)	Logs off the interactive user, shuts down the system.
	SuspendWindows (see page 123)	Suspends Windows
	TurnMonitorOff (see page 123)	Turns the monitor off
	TurnMonitorOn (see page 123)	Turns the monitor on
	UpdateWindowPosition (see page 124)	Updated the window position

1.4.10 SystemInfo.h

Routines for querying information about operating system

Functions

	Name	Description
	CurrentIPAddressA (see page 24)	Returns the IP address
	CurrentIPAddressW (see page 24)	Returns the IP address
	IsCitrixSession (see page 73)	Checks if application is running in Citrix session
	IsConnectedToInternet (see page 74)	Checks if PC is connected to Internet
	IsMediaCenter (see page 77)	Checks if the Media Center version of Windows is installed
	IsMetroActive (see page 77)	Checks if metro interface is active
	IsMultiTouchReady (see page 78)	Checks if the system is multi touch ready
	IsNativeWin64 (see page 78)	Checks if the application is native 64 bit executable
	IsRemoteSession (see page 78)	Checks if application is running in RDP session
	IsTabletPC (see page 79)	Checks if the application is running on the Tablet PC
	IsWindows10 (see page 82)	Checks if PC is running Windows 10 or better
	IsWindows7 (see page 82)	Checks if PC is running Windows 7 or better
	IsWindows8 (see page 82)	Checks if PC is Running Windows 8 or better
	IsWindowsVista (see page 82)	Checks if PC is running Windows Vista or better
	IsWindowsXP (see page 82)	Checks if PC is running Windows XP
	IsWindowsXPSP2 (see page 83)	Checks if PC is running Windows XP with Service Pack 2 installed
	IsWow64 (see page 83)	Checks if the 32 bit application runs on 64 bit Windows
	PortAvailable (see page 88)	Checks if the port with specified number is available
	RecycleBinEmpty (see page 100)	Returns TRUE if Windows Recyce Bin is empty

Index

A

B

C

D

E

F

G

H

I

L

M

N

P

Q

R

S

T

U

V

W