

SKI ActiveX Control Wrapper

Version 3.1

User's Guide

Issue 1.1

November 2006

Windows 9x / Me / NT / 2K / XP

Purpose

The purpose of this document is to explain how to use the SKI ActiveX Control Wrapper DLL. It describes the function calls and callback routines, plus general information about the how to use and build applications using the wrapper.

www.ScreenKeys.com



SKI ActiveX Control Wrapper

Information in this document is subject to change without notice.

The latest revisions of the ScreenKey documentation and software are placed on the SKI Web site.

Web: www.ScreenKeys.com

Technical Support is available

via Email support@ScreenKeys.com

via Web: www.ScreenKeys.com

© 2006 SK Interfaces Ltd.

All rights reserved.

DISCLAIMER:

SKI reserves the right to revise program interfaces, data file formats and functionality, at any time.

Foreword

The SKI ActiveX Control is a COM-control to be used to interface an application written in a COM-enabled programming language with a ScreenKey console, using a development environment utilizing the qualities of COM (ActiveX) controls, such as IntelliSense® and parameter enumeration. It is designed to work with languages like MS Visual Basic and MS Visual C++, using MS Visual Studio 6.0.

This document describes the Wrapper DLL, a module that provides access to the SKI ActiveX Control functionality for non-COM enabled programming languages, for example C etc.

This document describes the functions provided by the wrapper and how to use it from the C programming language.

Please refer to the “SKI ActiveX Control User’s Guide” for an in-depth description of the features and normal usage of the SKI ActiveX Control.

SKI – ScreenKey Interface

ScreenKey technology has been in use for a considerable number of years. In this time, SKI has developed a range of software tools to simplify the task of integrating a ScreenKey console into an application or hardware solution.

SKI Software Toolset (SKI) implements a new approach to controlling ScreenKey consoles and consoles. SKI is a toolset designed specifically for Windows 32-bit platforms and utilizes Microsoft COM technology.

SKI is particularly targeted at Application Developers, offering a host of developer tools to allow tight and dynamic integration of ScreenKey™ technology.

SKI implements a “SAC” (ScreenKey Active Control) approach to the control of a ScreenKey console. SAC files define how to control a ScreenKey console and include text, graphics, menus, menu navigations and keystroke return values. The SAC Engine and SAC Controller allow developers to integrate a ScreenKey console with minimal programming effort using SAC files.

Alternatively, the SKI ActiveX Control provides developers with the ability to directly control each key on a ScreenKey console. This is suitable for user interface that does not change significantly and/or ScreenKeys that have to display dynamic data, e.g. equipment temperature or an alarm condition. The SKI ActiveX Wrapper provides access to the SKI ActiveX Control features for non-COM enabled programming languages.

Table of Contents

- SKI ACTIVEX CONTROL OVERVIEW..... 5**
 - What is ScreenKey Console ?..... 5*
 - What is SKI ActiveX Control ? 5*
 - Who should read this document ?..... 5*
 - Requirements 6*
 - Cross reference..... 6*

- INSTALLATION PROCEDURE..... 7**
 - Hardware Installation 7
 - Software Installation..... 7
 - Uninstallation of Software..... 8
 - Redistributable Components / Files..... 9
 - Downloadable Console Firmware..... 10*
 - DCOM & Windows 95..... 10*

- TECHNICAL DESIGN 11**
 - SKI ActiveX Control 11
 - Wrapper DLL 12

- USAGE..... 13**

- WRAPPER DLL FUNCTIONS..... 15**
 - SetKeyPressEventCallBack 15*
 - SetKeyLockTurnEventCallBack..... 16*
 - SetMSRSwipeEventCallBack 17*
 - SetErrorEventCallBack 18*

- APPENDICES**

- LANGUAGE INTERFACES..... 19**
 - Standard C 19
 - C Windows programs 19*
 - DOS Console programs..... 20*
 - Methods 20*
 - Properties 20*
 - Events 21*
 - Other non-COM enabled languages 21

- DOCUMENTATION CONTROL..... 22**
 - A.1 Change Control..... 22*
 - A.2 Abbreviations Used/Terms of Reference 22*
 - A.3 Historical Change Reference 22*
 - A.4 Change Summary 23*

I N T R O D U C T I O N

SKI ActiveX Control Overview

What is ScreenKey Console ?

The ScreenKey console is an Input/Output device for use with Point-of-Sale (POS) and other specialized applications, such as process control, dealer desks, call centers etc. The ScreenKey console combines the best features of a standard keyboard and a TouchScreen. It is made up of conventional keys and special ScreenKeys. A ScreenKey acts like a conventional key but each ScreenKey has a built-in LCD display panel. This enables the ScreenKey's legend to be changed dynamically. The LCD display area is 32 pixels across by 16 pixels down.

The ScreenKey console connects to the PC via the Serial RS232 port.

What is SKI ActiveX Control ?

SKI ActiveX Control is a software package that allows the ScreenKey console to be easily integrated into Windows based Point-of-Sale applications, or other types of applications. It can be used by modern programming languages like Visual C++, Visual Basic etc. SKI ActiveX Control does not work with SAC files but offers control of individual ScreenKeys and their properties.



Who should read this document ?

This User's Guide is intended for the person who will integrate the ScreenKey console with an application. The programmer must be familiar with using 'dll' controls in general, the programming language, the development tools, the operating system, and know how the ScreenKey console works.

This guide assumes that the reader is familiar with all of the above, with Microsoft Windows and Windows commands. This guide should be read in conjunction with the "SKI ActiveX Control User's Guide".

Requirements

Most applications should be suitable for use with SKI ActiveX Wrapper as long as they meet the following requirements...

Operating System:	Windows 9x, Me, Nt, 2K, or XP
PC Hardware:	The application must run on standard PC hardware. One COM port must be available.
Available PC Resources:	The PC must have sufficient available resources to run the ScreenKey console software. The SKI ActiveX Wrapper can be used on a PC with the amount of memory required for running the operating system.

Cross reference

- [1] ScreenKey Getting Started & Installation Guide
- [2] SKI ActiveX Control User's Guide
- [3] ScreenKey Console Technical Reference Manual
- [4] ScreenKey Low-Level Interface Programmer's Guide

I N S T A L L A T I O N

Installation Procedure

Hardware Installation

See the **ScreenKey Console, Getting Started & Installation Guide** document for details of hardware installation.

Software Installation

The SKI ActiveX Control is supplied on CDROM as part of the SKI Software Toolset. The Wrapper is distributed separately upon request. Alternatively it may be downloaded as an individual component from the web (www.ScreenKeys.com).

The Wrapper uses the SKI ActiveX Control and requires that the SKI ActiveX Control **must** be installed **before** attempting to use the Wrapper.

When the SKI ActiveX Control is distributed by CDROM, the CD usually incorporates an autorun facility that launches an installation helper utility (e.g. HTML file). Follow the on-screen instructions as described by this utility. If the utility is not present or the autorun feature is disabled on your computer, you can install the software directly from the CD. The ScreenKey software package typically includes several software applications that reside separately on the CD. Find and open the SKI ActiveX Control sub-folder using Windows explorer from the root of the CD and run Setup.exe from this folder.

Start the **Setup.exe** program, and follow the on-screen instructions.

- The Software License Agreement has to be accepted before the installation can proceed.
- The default destination location for the software is **C:\Program Files\ScreenKey**, but this may be changed by clicking on the Browse button. The installation will build a folder structure from the destination location for the different parts of the software, the control will be put in `.\Bin`, the documentation in `.\Doc`, and the samples in `.\Samples`, etc.
- The user may specify which components to install individually.
- When the control has been installed, it (three files) will register in the registry. If registration goes well, the user has to click OK for the files `SkAxCtlps.dll` and `SkAxCtl.dll` (problems registering `SkAxCtl.exe` will be reported by the Setup).

If errors occur at this stage, i.e. the registering did not succeed, the Control will not be functioning. In this case, try the following:

1. Uninstall earlier versions of the control, and reinstall the control.
2. Reboot the PC, stop all applications and install again.
3. Log on as Administrator, and install again.
4. View the Windows Event Log, correct any errors and install again.
5. Contact support@ScreenKeys.com

If the control fails to register a separate batch file is supplied, Register.bat, which can be run from a DOS shell when errors have been corrected.

- To install the Wrapper, simply copy the three wrapper files (SkAxWrap.dll, SkAxWrap.lib and SkAxWrap.h) into the target project folder.

Uninstallation of Software

To uninstall the SKI ActiveX Control, do the following:

1. Terminate any application using the SKI ActiveX Control, and make sure no other application is using any of the other files, like documentation and sample code.
2. Start **Add/Remove Programs**, from the **Control Panel**.
3. Locate the line **SKI ActiveX Control** (or ScreenKey Control) in the list of installed programs.
4. Click the **Add/Remove** button.
5. Click **Yes** in the dialog if you want to completely remove the SKI ActiveX Control and all of its components.
6. The SKI ActiveX Control with all of its components, documentation and sample programs will then be removed, all folders will be deleted if they are empty, and all registry entries will be removed.
7. If parts of the control could not be removed, a button named Details... will be visible, and the user can click on this to see what was not successfully removed.



Any changed files, like sample code etc., will be left, and if the control was in use by an application it will not be possible to remove this.

8. The software should now have been removed, and **Add/Remove Programs** can be terminated.

Redistributable Components / Files

When using the SKI ActiveX Control and wrapper in an application some files have to be redistributed. They should all reside in the same folder to avoid problems. The application may be distributed with the following files:

File name	Description	Registering
SkAxCtl.exe	SKI ActiveX Control. This file is the control itself, and is ALWAYS necessary.	YES
SkAxCtlps.dll	Proxy Stub for SKI ActiveX Control. This file is necessary if SkAxCtl.dll is in use.	YES
SkAxCtl.dll	SKI ActiveX Control Interface. This file is necessary if the control is used from e.g. Visual Basic, or Visual C++ with the control running in the applications process space.	YES
*.skf	ScreenKey Font files. These files are distributed for CodePages other than 1252. They should be distributed with the control, residing in the same directory as SkAxCtl.exe.	N/A
SkAxWrap.dll SkAxWrap.lib	SKI ActiveX Control Wrapper DLL. If the Wrapper DLL is used, the SKI ActiveX Control Interface (SkAxCtl.dll) is not necessary. This file has to be installed either in the folder where the application using it resides, or in the system folder, or in a folder mentioned in the system path. The SkAxWrap.lib is the import library for the Wrapper DLL.	NO
SkAxWrap.dll	SKI ActiveX Wrapper	

Some of the EXE and DLL files have to be registered in the registry. SkAxCtl.exe can register itself, but the DLL files need to be registered with the program RegSvr32.exe. RegSvr32.exe is distributed from Microsoft Corp., and can be used freely. It can be found in the Windows System folder (WinNt\System32 or Windows\System).

To register the components one has to execute the following commands, in the following order:

```
RegSvr32 SkAxCtlPS.Dll

SkAxCtl.exe /RegServer

RegSvr32 SkAxCtl.Dll
```

This can be done by executing a batch file, or by including the commands in an installation program.

Be aware that if both files, Regsvr32.exe and the DLLs, are not in the current working folder or the RegSvr.exe program is not in the standard path, it might be necessary to specify a path for the either one or both files.

Note that the components can be installed in any suitable directories, as long as all supplementary files are installed together with SkAxCtl.exe, and that they are correctly registered.

Downloadable Console Firmware

The ScreenKey console supports downloadable firmware as a means of adding new functionality and/or correcting software bugs. SKI will issue new firmware from time to time.

The path and name of this firmware is defined in the Windows registry. The installation of the SKI ActiveX Control automatically creates the following registry key:

HKEY_LOCAL_MACHINE\SOFTWARE\ScreenKeys\ScreenKey ActiveX Control\ROMFullPath

Set the value of this key to point to a new firmware download file, e.g.:

“C:\Program Files\ScreenKey\Bin\PLL4_2.bin” for products such as the OEM-5400

or

“C:\Program Files\ScreenKey\Bin\SKC1_0.bin” for SK-7000 products

Set the value of this key to NULL (empty string) if there is no firmware file to download.

DCOM & Windows 95

The controls use COM to communicate with each other. COM is built in to all Windows operating systems, but there is a known error in COM with Windows 95. Before the controls can be installed on a Windows 95, DCOM95 has to be updated. This installation is available, free of charge, from Microsoft Corp., and is also available on the SKI Installation CD.

Technical Design

The SKI ActiveX Control is designed for use with different programming languages, and is split into two main modules:

- SKI ActiveX Control Interface (for COM-enabled programming languages only), or SKI ActiveX Wrapper (for non-COM enabled languages).
- SKI ActiveX Control

The **SKI ActiveX Control Interface** (or **SKI ActiveX Wrapper**) runs in the application's own process space, and therefore accessible from Visual Basic (or C, etc). The **Interface** DLL file is called **SkAxCtl.dll** and the Wrapper DLL file is called **SkAxWrap.dll**.

The **SKI ActiveX Control** itself is running in a separate process space, and therefore cannot be accessed directly from Visual Basic. It is accessible from C++ when it is created as an object. The control is an EXE file, called **SkAxCtl.exe**.

A Proxy Stub DLL is also necessary, this is called **SkAxCtlps.dll**.

A full description of the technical design of the SKI ActiveX Control is given in the SKI ActiveX Control User's Guide document.

SKI ActiveX Control

The SKI ActiveX Control consists of two modules, a client and a server. There may only be one instance each of the client and the server. The server handles the actual interface to the ScreenKey console.

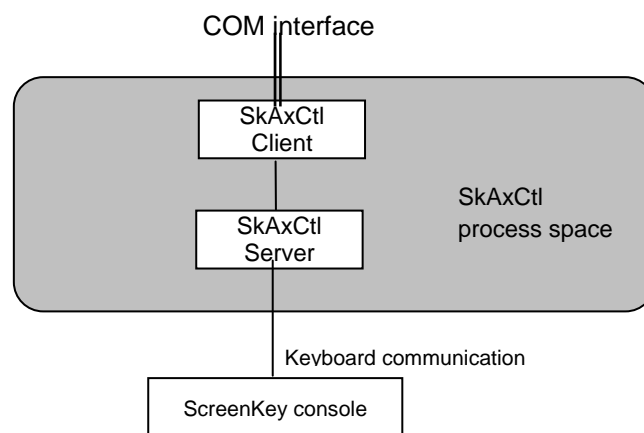


Figure 1 – SKI ActiveX Control

The use of a client and a server was intended to provide the opportunity to manage multiple applications interfacing to the same ScreenKey console, i.e. multiple SkAxCtl clients interfacing to one SkAxCtl server. However, this feature was removed in release 2.0 as it proved excessive to user requirements and increased the overall complexity and footprint of the SKI ActiveX Control unnecessarily.

Wrapper DLL

The Wrapper DLL for programming languages not supporting the COM interface is called SkAxWrap.dll. This DLL file can be linked to e.g. a C program, like any DLL library. It interfaces to the SkAxCtl.exe client/server control, and offers access to methods, properties (as function calls), and also implements events as callback functions. A header file is available, for inclusion into the C files.

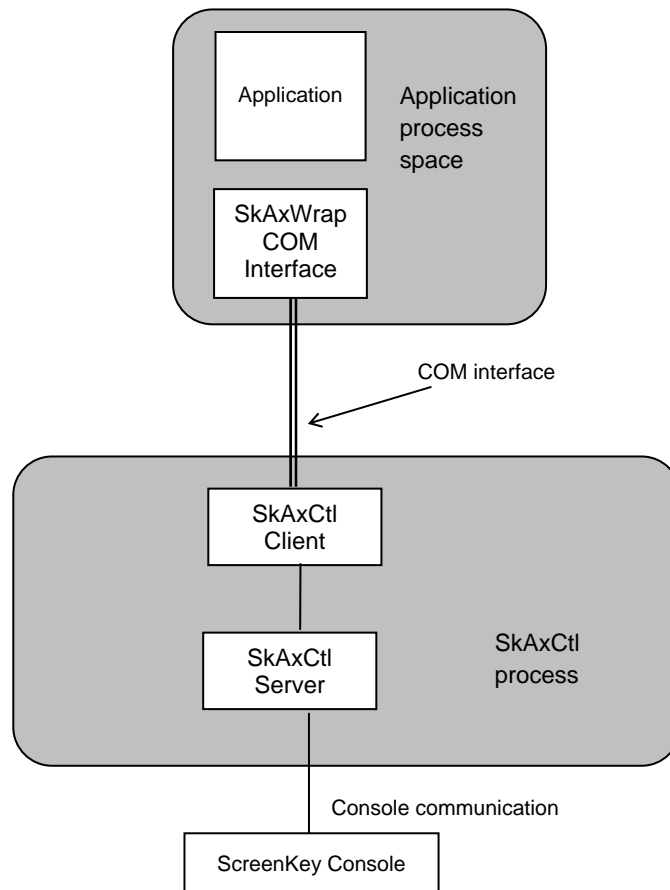


Figure 2 - Wrapper DLL, from e.g. C

U S A G E

Usage

The SKI ActiveX Control has different categories of properties, methods and events, and some of these have to be executed in a certain sequence. The detailed description of each method, property and event is described in the SKI ActiveX Control User's Guide.

The Wrapper provides a non-COM program access to the functionality of the SKI ActiveX Control. This document assumes the reader is familiar with this functionality.

The SKI ActiveX Control offers the following methods, properties and events:

Methods	Properties	Events
OpenKeyboard	KbdLayout	KeyPressEvent
CloseKeyboard	ErrorMode	KeyLockTurnEvent
DefineLargeKeytop	ErrorNumber	MSRSwipeEvent
RedefineKey	ErrorDescription	ErrorEvent
ConfigSleep	KeyPress	
ConfigFlash	KeyPressRow	
ConfigScroll	KeyPressCol	
ConfigRepeat	KeyLockPos	
SetKeyAttribute	MSRTracks	
SetCharSet	MSRTrack1Data	
DisplayText	MSRTrack2Data	
DisplayTextOnGraphics	MSRTrack3Data	
DisplayGraphics		
DisplaySimpleText		
SetLED		
Sound		
SendSimpleCommand		
RequestKbdData		
WriteTextODA ¹		
WriteTextTDA ¹		
WriteTextCDA ¹		
SelectCDA ¹		
SelectTDA ¹		
WriteGraphicTDA ¹		

¹ These methods are only available on the SK-7000 console.

The Wrapper DLL adds the following event-handling functions:

SetKeyPressEventCallBack	Set address of the key press event callback function.
SetKeyLockTurnEventCallBack	Set address of the KeyLock turn event callback function.
SetMSRSwipeEventCallBack	Set address of the MSR swipe event callback function.
SetErrorEventCallBack	Set address of the error event callback function.

These functions provide a mechanism for non-COM languages to provide callback functions references to handle the various console events.

Wrapper DLL functions

SetKeyPressEventCallback

Description

This method informs the Wrapper DLL of the address of the function handling key press events.

Parameters

fnKeyPress

Type: lpLong

The address of the function handling key press events.

The type lpLong is defined in SkAxWrap.h

Return Value None

Use after None

Remarks

Since the Wrapper DLL is meant to be used from languages not supporting COM (like C), and these does not have a standardized way of handling events, the application needs to avail of a function to handle these events. The Wrapper DLL needs to know the address of this function so it can be called when an event occurs. The function for handling key press events needs to be as below. Note that the name of the function can be different from the example below.

```
void KeyPressEvent( long Key )
{
    . . .
}
```

To inform the Wrapper DLL about the function do the following call:

```
SetKeyPressEventCallback( KeyPressEvent );
```

If no address is given (or NULL) this event will not be forwarded to the application.

See KeyPressEvent in the SKI ActiveX Control User's Guide for more information on callback (event) parameters.

SetKeyLockTurnEventCallback

Description

This method informs the Wrapper DLL of the address of the function handling KeyLock turn events.

Parameters

fnKeyLock

Type: lpLong

The address of the function handling KeyLock turn events.

The type lpLong is defined in SkAxWrap.h

Return Value None

Use after None

Remarks

Since the Wrapper DLL is meant to be used from languages not supporting COM (like C), and these does not have a standardized way of handling events, the application needs to avail of a function to handle these events. The Wrapper DLL needs to know the address of this function so it can be called when an event occurs. The function for handling KeyLock turn events needs to be as below. Note that the name of the function can be different from the example below.

```
void KeyLockEvent( long Key )
{
    . . .
}
```

To inform the Wrapper DLL about the function do the following call:

```
SetKeyLockTurnEventCallback( KeyLockEvent );
```

If no address is given (or NULL) this event will not be forwarded to the application.

See KeyLockTurnEvent in the SKI ActiveX Control User's Guide for more information on callback (event) parameters.

SetMSRSwipeEventCallback

Description

This method informs the Wrapper DLL of the address of the function handling MSR swipe events.

Parameters

fnNoMSRTrack

Type: lpLong

The address of the function handling MSR swipe events.

The type lpLong is defined in SkAxWrap.h

Return Value None

Use after None

Remarks

Since the Wrapper DLL is meant to be used from languages not supporting COM (like C), and these does not have a standardized way of handling events, the application needs to avail of a function to handle these events. The Wrapper DLL needs to know the address of this function so it can be called when an event occurs. The function for handling MSR swipe events needs to be as below. Note that the name of the function can be different from the example below.

```
void MSREvent( long NoTracks )
{
    . . .
}
```

To inform the Wrapper DLL about the function do the following call:

```
SetMSRSwipeEventCallback( MSREvent );
```

If no address is given (or NULL) this event will not be forwarded to the application.

See MSRSwipeEvent in the SKI ActiveX Control User's Guide for more information on callback (event) parameters.

SetErrorEventCallback

Description

This method informs the Wrapper DLL of the address of the function handling error events.

Parameters

fnErrorID

Type: lpCharLong

The address of the function handling error events.

The type lpCharLong is defined in SkAxWrap.h

Return Value None

Use after None

Remarks

Since the Wrapper DLL is meant to be used from languages not supporting COM (like C), and these does not have a standardized way of handling events, the application needs to avail of a function to handle these events. The Wrapper DLL needs to know the address of this function so it can be called when an event occurs. The function for handling error events needs to be as below. Note that the name of the function can be different from the example below.

```
void ErrorEvent ( char *ErrorText, long ErrorNo )
{
    . . .
}
```

To inform the Wrapper DLL about the function do the following call:

```
SetErrorEventCallback( ErrorEvent );
```

If no address is given (or NULL) this event will not be forwarded to the application.

See ErrorEvent in the SKI ActiveX Control User's Guide for more information on callback (event) parameters.

Language Interfaces

The Wrapper may be used from different programming languages. The syntax depends on the language, and using it with “C” is described below.

Standard C

Standard C does not support the COM interface. To use the SKI ActiveX Control from C, the Wrapper DLL has to be used. This is a standard Dynamic Link Library (DLL), which connects to the SKI ActiveX Control (SkAxCtl.exe). The Wrapper DLL replaces the SKI ActiveX Control Interface, it is not an addition to it – overhead is therefore not as bad might be expected. The overhead is actually less than from e.g. VB, since it in this case is only one RPC level (COM-interface).

The Wrapper DLL is generated for Visual C++ 6.0 (Visual Studio 6.0) compiler.

WARNING!

Because of a conflict with an existing Windows function, the property `ErrorMode` (from C++ `SetErrorMode`), has been renamed to **SetErrMode** in the Wrapper DLL. The reason this was not necessary in the ActiveX Control is that there it is a member of a class, and therefore unique.

C Windows programs

A Windows program written in plain C, like for Windows 3.1, can be compiled with the C-compiler or the C++ compiler. The Wrapper DLL supports both of these approaches, by exporting two sets of functions with different decoration. The program can be compiled with either the `/TC` (compile as if a .c file) or `/TP` (compile as if a .cpp file) option.

A header file (`SkAxWrap.h`) is available for inclusion in the code. This contains all required definitions, type-definitions and function prototypes, and should be included in any file accessing the wrapper functions.

The `implib` file `SkAxWrap.lib` must be specified as an Object/library module in the linker option dialog.

Be aware that classic C does not have any exception handling, and the application therefore has to check the return values after each function invoked. The `ErrorMode` property is therefore not as in C++ or Visual Basic, it only sets the FATAL bit into the error codes (oring in `FATAL_ERROR_BIT`). If the `ErrorMode` is in use the application can detect if a fatal error has occurred by checking this bit. If the error code is to be used, e.g. to check against predefined error codes (enumerations) this bit must be mapped out first. If an error occurs the error code will always be set to the corresponding error code, independently of the current error mode. The only difference is that the FATAL bit will be set if the error is in the specified categories.

DOS Console programs

The Wrapper DLL will also work with Win32 Console Applications written in C. A Console application is normally used to make DOS programs into Windows compatible programs. A DOS program can normally be recompiled without major changes into a Console application.

This way it is possible to add support for ScreenKey consoles in an old DOS application using the SKI ActiveX Control.

Methods

All methods return a *HRESULT* value, which contains the status (error number). If the program is a NON-Windows program, the return value is *long*.

Example:

```
char str[128];
RetVal = OpenKeyboard( "COM1", KBD_TYPE_2000,
                     EXCLUSIVE_USE | FAIL_TEXT_TOO_LONG );
if ( RetVal ) {
    GetErrorDescription( Str );
    printf( "OpenKeyboard: %lx %s\n", RetVal, Str );
}
```

Properties

Properties are implemented as functions. They also return a *HRESULT* value, which contains the status (error number). In-properties have the prefix *Get*, and take a pointer to the actual type as parameter. Out-properties have the prefix *Set*, and take a value of the actual type as parameter.

Example:

```
void Function( void )
{
    MSR_TRACKS MSRTracksRead;

    SetErrorMode( ERRMODE_NONCRITICAL );

    . . .

    GetMSRTracks( &MSRTracksRead );
}
```

NOTE:

The property *ErrorMode* (*SetErrorMode*) is named *SetErrMode* in the Wrapper DLL (see above).

Events

Events must be handled specially because there is no support for this in standard C. A function has to be created, in a certain fashion. Then the address of this has to be given to the Wrapper DLL, which again will call the function at this address when an event occurs. The event functions are the following:

```
void KeyPressEvent( long Key);
void KeyLockTurnEvent( long Key );
void MSRSwipeEvent( long NoTracks );
void ErrorEvent( char *ErrorText, long ErrorNo );
```

These prototypes are defined in the header file, but they may be named differently, but then the application will have to define the prototype for these on its own.

To define the call-back functions to the Wrapper DLL the corresponding functions have to be called:

```
SKAXWRAP_API void SetKeyPressEventCallBack( lpLong fnKeyPress );
SKAXWRAP_API void SetKeyLockTurnEventCallBack( lpLong fnKeyLock );
SKAXWRAP_API void SetMSRSwipeEventCallBack( lpLong fnNoMSRTrack );
SKAXWRAP_API void SetErrorEventCallBack( lpCharLong fnErrorID );
```

The types used are all defined in the included header file (SkAxWrap.h).

Example on implementation of KeyPress call-back:

```
. . . . .
// KeyPressEvent is the name of the function defined in module.
SetKeyPressEventCallBack( KeyPressEvent );
. . . . .

void KeyPressEvent( long KeyNo )
{
    char TempStr[256];

    // We have received a keypress event from control
    sprintf( TempStr,
             "Key Press Event from ScreenKey Keyboard \r\nValue: %d",
             KeyNo );
    MessageBox( NULL, TempStr, "KEY PRESS", MB_OK );
}
```

Other non-COM enabled languages

Other non-COM-enabled languages can also use the SKI ActiveX Control, as long as they can use a standard DLL, by using the Wrapper DLL. The include file (or similar) has to be created by the developer, and the sequence of parameter has to be done right!

D O C U M E N T A T I O N C O N T R O L

Documentation Control

A.1 Change Control

This document is the responsibility of the author and is subject to formal change control after the initial approved release (i.e. issue 1.0).

A.2 Abbreviations Used/Terms of Reference

ScreenKey	Registered Trademarked key-switch with a backlit LCD screen incorporated.
ScreenKey console	SKI's range of consoles containing ScreenKeys.
PASKeyboard	Former name of ScreenKey console
ActiveX	Name of technology used for communication between applications/modules.
COM	As ActiveX.
BSTR	Visual Basic type string.
LED	Light Emitting Diode.
*MSR	Magnetic Stripe Reader
POS	Point of Sale--the cash register in a shop.

A.3 Historical Change Reference

Issue	Date	Author	Changes Made
1.0	07/04/04	M McDonnell	First release
1.1	30/11/06	M McDonnell	Update for new registry path

A.4 Change Summary

Issue	Change description
1.1	Update with new registry path for download firmware and handshake info, remove MSR Utility references