Registry Functions

Introduction

The *registry* is a system-defined database that applications and Microsoft® Windows® system components use to store and retrieve configuration data.

This appendix to the W/32 App Builder User's Manual describes functions that may be called from a W/32 program to access and modify registry data. It is assumed that the programmer understands the purpose of the registry, and the basics of its use in Windows operating systems.

Release Level

The registry functions defined below are available in W/32 App Builder Runtime Systems and W/32 App Builder Development Systems with version numbers at or above 9.22G.

List of Registry Functions

The following functions have been added to the set of callable functions:

cRegOpenKey

cRegCloseKey

cRegSetValue

cRegDeleteValue

cRegDeleteKey

cRegQueryValue

cRegInfoKey

Overview of Registry Functions

The W/32 registry functions access the system registry by specifying, opening and operating on registry keys. Each of the W/32 registry functions must be supplied with a **key handle** as the first parameter, and will return a return code (passed as an ADDOROF(<int variable>)) for the function **result** as last parameter.

Each of the W/32 registry functions has a syntax of the form:

CALL cReg…(key%, …, ADDROF(result%))

where

key% is either the key handle returned by previous call to cRegOpenKey() or one of following predefined values:

HKEY.CLASSES.ROOT% = 80000000H

HKEY.CURRENT.USER% = 80000001H

HKEY.LOCAL.MACHINE% = 80000002H

HKEY.USERS% = 80000003H

Result% is a variable supplied for the return result. The return result will be negative, positive or zero with the following meaning for each type of return result:

A result of 0 means that the function was successful.

A result of -1 is returned by cRegOpenKey when it successfully creates the registry key.

A positive result (>0) means that the function failed, and the value or **result%** will contain the Windows API error code.

Syntax and Operation of Registry Functions

The following section lists each of the currently supported registry functions and describes their specific behavior. Each of these functions must be invoked by means of a W/32 CALL statement.

cRegOpenKey(key%, subkey$, ADDROF(subkey%), class$, ADDROF(result%))

The cRegOpenKey function opens or creates a subkey of opened or predefined key. The parameters are:

subkey$ subkey name to open

subkey% subkey handle returned by the function

class$ subkey class for a new subkey or 0

When a 0 is passed instead **class$** name, the function opens the existing subkey or fails if the subkey does not exist.

When a string is passed as the **class$** parameter, the function opens an existing subkey or creates a new one.

The function returns a –1 in the **result%** variable if the key was successfully created, or 0 if an existing subkey was opened.

cRegCloseKey(key%, ADDROF(result%))

The cRegCloseKey function closes the key handle.

cRegSetValue(key%,val$, type%, {int% | str$ | MATADDROF(strarr$) }, ADDROF(result%))

The cRegSetValue function assigns a value to the key. The parameters are:

val$ the key value name

type% the key value type as described in the table below:

Symbolic name	Val	Registry key type	4th parameter type
REG.SZ%	1	ASCII string	STRING ($)
REG.EXPAND.SZ%	2	ASCII string with env. variables	STRING ($)
REG.BINARY%	3	Binary	STRING ($)
REG.DWORD%	4	Double word	INT (%)
REG.DWORD.BIG.ENDIAN%	5	Double word with reversed byte order	INT (%)
REG.MULTI.SZ%	7	Multi-string	STRING ARRAY ($)

Int% | str$ | MATADDROF(strarr$) is the value to which the the key should be set, and can be integer or string variable or string array (see table above).

cRegDeleteValue(key%, value$, ADDROF(result%))

Deletes the value with given name (**value$**) from the key whose handle is **key%**.

cRegDeleteKey(key%, subkey$, ADDROF(result%))

Deletes the subkey (**subkey$**) from the key whose handle is **key%**.

cRegQueryValue(key%, value$, { ADDROF(int%) | ADDROF(str$) | MATADDROF(strarr$) } , ADDROF(result%))

For a given value **value$** of a key whose handle is **key%,** returns into a variable passed as the third parameter. The third paramater may be either integer, string, or an array of strings, and the function fails if the type of third parameter does not match the registry key type.

cRegInfoKey(key%, ADDROF(class$), ADDROF(subkey.count%), ADDROF(value.count%), ADDROF(result%))

For a given key whose handle is **key%**, returns the class name of key% and number of subkeys and values in **subkey.count%** and **value.count%.**

Sample W/32 Program

The following program shows how the registry functions may be used to define, query, and delete a registry subkey:

```
HKEY.CLASSES.ROOT%  = 80000000H
HKEY.CURRENT.USER%  = 80000001H
HKEY.LOCAL.MACHINE% = 80000002H
HKEY.USERS%         = 80000003H

REG.SZ%               = 1
REG.EXPAND.SZ%        = 2
REG.BINARY%           = 3
REG.DWORD%            = 4
REG.DWORD.BIG.ENDIAN% = 5
REG.MULTI.SZ%         = 7
```

```
call cRegOpenKey( HKEY.LOCAL.MACHINE%, "NewSubKey", ADDROF(key%), "NewClass",
ADDROF(result%) )
print "cRegOpenKey() : Open NewSubKey = "; result%
wait

call cRegSetValue( key%, "String", REG.SZ%, "String value", ADDROF(result%) )
print "cRegSetValue() : Set SZ value = "; result%
wait

call cRegSetValue( key%, "Dword", REG.DWORD%, 1234, ADDROF(result%) )
print "cRegSetValue() : Set DWORD value = "; result%
wait

dim msz$(1,3)
msz$(0,0) = "String 1 1"
msz$(0,1) = "String 1 2"
msz$(0,2) = "String 1 3"
msz$(0,3) = "String 1 4"
msz$(1,0) = "String 2 1"
msz$(1,1) = "String 2 2"
msz$(1,2) = "String 2 3"
msz$(1,3) = "String 2 4"

call cRegSetValue( key%, "MultiSz", REG.MULTI.SZ%, MATADDROF(msz$),
ADDROF(result%) )
print "cRegSetValue() : Set MULTI_SZ value, RC= "; result%
wait

call cRegInfoKey( key%, ADDROF(class$), ADDROF(subkey.count%),
ADDROF(value.count%), ADDROF(result%) )
print "cRegInfoKey() : Set DWORD value, RC= ";result%; " class='"; class$; "'"; "
sukeys="; subkey.count%; " values="; value.count%
wait

call cRegQueryValue( key%, "Dword", ADDROF( s% ), ADDROF(result%)  )
print "cRegQueryValue() : Query DWORD value, RC= "; result%; "  dword="; s%
wait

call cRegQueryValue( key%, "String", ADDROF( s$ ), ADDROF(result%)  )
print "cRegQueryValue() : Query SZ value, RC= "; result%; "  str='"; s$; "'"
wait

dim a$(2)
call cRegQueryValue( key%, "MultiSz", MATADDROF( a$ ), ADDROF(result%)  )
print "cRegQueryValue() : Query MULTI_SZ value, RC= "; result%; "  arr="; a$(0);
";"; a$(1); ";"; a$(2)
wait

call cRegDeleteValue( key%, "Dword", ADDROF(result%)  )
print "cRegDeleteValue() : Delete DWORD value,RC= "; result%
wait

call cRegDeleteValue( key%, "String", ADDROF(result%)  )
print "cRegDeleteValue() : Delete SZ value, RC= "; result%
wait

call cRegDeleteValue( key%, "MultiSz", ADDROF(result%)  )
print "cRegDeleteValue() : Delete MULTI_SZ value, RC= "; result%
wait

call cRegCloseKey( key%, ADDROF(result%) )
print "cRegCloseKey() : Close subkey, RC= "; result%
wait
```

```
call cRegDeleteKey( HKEY.LOCAL.MACHINE%, "NewSubKey", ADDROF(result%)  )
print "cRegDeleteKey() : Delete NewSubKey, RC= "; result%
wait
```