# 软件使用次数的限制

为限制软件使用次数，编写了一个CSecurity类。这里说明类的使用。

该类有三个变量：

CString MainPath; //程序在注册表中的路径

CString KeyName; //键值项名

int InitNum; //最大使用次数

第一个参数是程序主键在注册表主键HKEY\_LOCAL\_MACHINE下的路径，第二个参数是键值项的名称，第三个参数是初使用设置的使用计数，即最大使用次数，使用时在程序中定义全局变量，并在初始化函数中对该类对象进行初始化。

例如：

sy.InitNum=100;

sy.MainPath="SOFTWARE\\ReadWriteTimes";

sy.KeyName="Times";

该类有两个函数接口：

int Readtimes();

void Writetimes(int num);

第一个函数是读取使用计数，该函数第一次使用时会在注册表中写入程序注册表中相应路径下创建相差键，并设置键值。

第二个函数是写入使用计数。

注册表中键值的读写都经过加密和解密处理。

头文件

// Security.h: interface for the CSecurity class.

//

//////////////////////////////////////////////////////////////////////

#if !defined(AFX\_SECURITY\_H\_\_96D62D30\_14A2\_4016\_8E79\_14BDD9A25DA7\_\_INCLUDED\_)

#define AFX\_SECURITY\_H\_\_96D62D30\_14A2\_4016\_8E79\_14BDD9A25DA7\_\_INCLUDED\_

#if \_MSC\_VER > 1000

#pragma once

#endif // \_MSC\_VER > 1000

class CSecurity

{

public:

CSecurity();

virtual ~CSecurity();

int Readtimes();

void Writetimes(int num);

private:

CString Encrypt(CString S, WORD Key); // 加密函数

CString Decrypt(CString S, WORD Key); // 解密函数

public:

CString MainPath; //程序在注册表中的路径

CString KeyName; //键值项名

int InitNum;

};

#endif // !defined(AFX\_SECURITY\_H\_\_96D62D30\_14A2\_4016\_8E79\_14BDD9A25DA7\_\_INCLUDED\_)

源文件

// Security.cpp: implementation of the CSecurity class.

//

//////////////////////////////////////////////////////////////////////

#include "stdafx.h"

#include "ReadWriteTimes.h"

#include "Security.h"

#include <ATLBASE.H>

#ifdef \_DEBUG

#undef THIS\_FILE

static char THIS\_FILE[]=\_\_FILE\_\_;

#define new DEBUG\_NEW

#endif

#define C1 52845

#define C2 22719

#define C3 65

//////////////////////////////////////////////////////////////////////

// Construction/Destruction

//////////////////////////////////////////////////////////////////////

CSecurity::CSecurity()

{

InitNum=100;

MainPath="";

KeyName="";

}

CSecurity::~CSecurity()

{

}

//加密算法

CString CSecurity::Encrypt(CString S, WORD Key) // 加密函数

{

CString Result,str;

int i,j;

Result=S; // 初始化结果字符串

for(i=0; i<S.GetLength(); i++) // 依次对字符串中各字符进行操作

{

Result.SetAt(i, S.GetAt(i)^(Key>>8)); // 将密钥移位后与字符异或

Key = ((BYTE)Result.GetAt(i)+Key)\*C1+C2; // 产生下一个密钥

}

S=Result; // 保存结果

Result.Empty(); // 清除结果

for(i=0; i<S.GetLength(); i++) // 对加密结果进行转换

{

j=(BYTE)S.GetAt(i); // 提取字符

// 将字符转换为两个字母保存

str="12"; // 设置str长度为2

str.SetAt(0, C3+j/26);//这里将65改大点的数例如256，密文就会变乱码，效果更好，相应的，解密处要改为相同的数

str.SetAt(1, C3+j%26);

Result += str;

}

return Result;

}

//解密算法

CString CSecurity::Decrypt(CString S, WORD Key) // 解密函数

{

CString Result,str;

int i,j;

Result.Empty(); // 清除结果

for(i=0; i < S.GetLength()/2; i++) // 将字符串两个字母一组进行处理

{

j = ((BYTE)S.GetAt(2\*i)-C3)\*26;//相应的，解密处要改为相同的数

j += (BYTE)S.GetAt(2\*i+1)-C3;

str="1"; // 设置str长度为1

str.SetAt(0, j);

Result+=str; // 追加字符，还原字符串

}

S=Result; // 保存中间结果

for(i=0; i<S.GetLength(); i++) // 依次对字符串中各字符进行操作

{

Result.SetAt(i, (BYTE)S.GetAt(i)^(Key>>8)); // 将密钥移位后与字符异或

Key = ((BYTE)S.GetAt(i)+Key)\*C1+C2; // 产生下一个密钥

}

return Result;

}

int CSecurity::Readtimes()

{

CRegKey RegKey;

LPCSTR lpPath=MainPath;

int num=0;

if (RegKey.Open(HKEY\_LOCAL\_MACHINE,lpPath,KEY\_READ|KEY\_WRITE)==ERROR\_SUCCESS)

{

//打开注册表成功，设置键值

TCHAR str[100];

DWORD pCount=100;

if(ERROR\_SUCCESS==RegKey.QueryValue(str,KeyName,&pCount))

{

//解密

CString NewValue(str);

NewValue=Decrypt(NewValue,137);

num=\_ttoi(NewValue);

NewValue.Format("%d",num-1);

//加密

NewValue=Encrypt(NewValue,137);

RegKey.SetValue(NewValue,KeyName);

}

else

{

return -1;

}

}

else

{

RegKey.Create(HKEY\_LOCAL\_MACHINE,lpPath,REG\_NONE,

REG\_OPTION\_NON\_VOLATILE,KEY\_ALL\_ACCESS,NULL,NULL);

CString str;

str.Format("%d",InitNum);

//加密

str=Encrypt(str,137);

RegKey.SetValue(str,KeyName);

num=InitNum;

}

RegKey.Close();

return num;

}

void CSecurity::Writetimes(int num)

{

// TODO: Add your control notification handler code here

CRegKey RegKey;

LPCSTR lpPath=MainPath;

CString strNum;

strNum.Format("%d",num);

if (RegKey.Open(HKEY\_LOCAL\_MACHINE,lpPath,KEY\_READ|KEY\_WRITE)==ERROR\_SUCCESS)

{

//打开注册表成功，设置键值

//加密

strNum=Encrypt(strNum,137);

RegKey.SetValue(strNum,KeyName);

}

RegKey.Close();

}