ЛЕКЦИЯ. РЕЕСТР WINDOWS

Реестр – *это централизованное хранилище системных настроек*.
Управляется подсистемой **Configuration Manager** ядра операционной системы.

Реестр играет ключевую роль в конфигурации систем Windows и в управлении ими. Он является хранилищем как для общесистемных настроек, так и для настроек для каждого пользователя. Хотя многие считают реестр некими статичными данными, хранящимися на жестком диске, реестр также является окном в различные находящиеся в памяти структуры, обслуживаемые исполняющей системой и ядром Windows.

Начнем с обзора структуры реестра, рассмотрения поддерживаемых им типов данных и краткого изучения ключевой информации, которую Windows содержит в реестре. Затем заглянем во внутренности диспетчера конфигурации — компонента исполняющей системы, ответственного за реализацию базы данных реестра. Мы также рассмотрим внутреннюю дисковую структуру реестра, порядок извлечения конфигурационной информации по запросам приложений и меры защиты этой важной системной базы данных.

Шесть корневых разделов

HKEY_CURRENT_USER Хранит данные, связанные с текущим пользователем, вошедшим в систему

HKEY_USERS Хранит информацию обо всех учетных записях, имеющихся на машине

HKEY_CLASSES_ROOT Хранит файловые связи и информацию о регистрации объектов, относящихся к модели компонентных объектов — Component Object Model (COM)

HKEY_LOCAL_MACHINE Хранит информацию, связанную с системой

HKEY_PERFORMANCE_DATA Хранит информацию о производительности

HKEY_CURRENT_CONFIG Хранит определенную информацию о текущем профиле

Состоит из следующих корневых узлов:

HKey_Classes_Root (HKCR)

HKey_Current_User (HKCU)

HKey_Local_Machine (HKLM)

HKey_Users (HKU)

HKey_Current_Config (HKCC)

HKEY_DYN_DATA (HKDD) (Win9x Only)

Раздел HKCR состоит из трех типов информации:

- ассоциации с расширениями файлов;
- регистрации COM-классов
- виртуализированный корневой раздел реестра системы для управления учетными записями пользователей — User Account Control

Для каждого зарегистрированного расширения имени файла есть свой раз-

дел. Большинство разделов содержат параметр типа REG_SZ, который указывает на другой раздел в HKCR, содержащий информацию, связанную с тем классом файлов, который представляет данное расширение.

Например, HKCR\.xls будет указывать на информацию о файлах Microsoft

Office Excel в таком разделе, как HKCU\.xls\Excel.Sheet.8. В других разделах

содержатся подробности конфигурации для COM-объектов, зарегистрированных в системе. Виртуализированный реестр UAC находится в разделе VirtualStore, который не имеет никакого отношения к другим разновидностям данных, хранящихся в HKCR.

How Many Processors Does Your O/S Support

a. HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\RegisteredProcessors

(Windows 2000)

Check the registry value to see how many processors your version of Windows supports.

This registry key value is informational only. In other words, changing the value to 128 won't enable your version of Windows to support 128 processors.

b. Remove Shutdown from the Start Menu

HKEY_CURRENT_USER\Software\Microsoft\Windows\Current Version\Policies\Explorer\NoClose

c. Power Down the Computer after Shutdown

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\PowerdownAfterShutdown

This REG_SZ or String value when set to 1 will automatically power off your computer if the BIOS supports this. A value of 0 will leave you at the "It is now safe..." screen.
This String Value set to 1 removes the Shutdown option from the Startmenu. Change it to 0 to allow shutdowns.

d. Restrict Users from Using GIFs and JPGs as Desktop Wallpaper
HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Policies \ActiveDesktop\NoHTMLWallPaper

To restrict users from using GIFs and JPGs as wallpaper (restrict them to BMP images) create or set this value to 1. A value of 0 will allow users to use all 3 types of images.

e. Turn off Low Disk Space Notification
(Windows XP)

HKEY_Current_User\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer\NoLowDiskSpaceChecks

This DWORD value set to 1 will disable the low disk space notification. 0 will enable it.

При установке приложения, включающего в себя службу, программа установки должна зарегистрировать службу в системе. Для регистрации службы программа установки вызывает Windows-функцию CreateService, связанную со службами функцию, которая находится в библиотеке Advapi32.dll

(%SystemRoot%\System32\Advapi32.dll). В Advapi32 (DLL-библиотеке улуч-

шенных API-функций — «Advanced API») реализованы все API-функции SCM,выполняемые на стороне клиента.

Когда программа установки регистрирует службу, вызывая функцию

CreateService, SCM на той машине, где будет располагаться служба, отправляется сообщение. После этого SCM создает для службы раздел реестра в разделе HKLM\SYSTEM\CurrentControlSet\Services. Раздел Services является долговременным представлением базы данных SCM. Отдельные разделы для каждой службы определяют путь к исполняемому образу, в котором содержится служба, а также параметры и настройки конфигурации.

После создания службы установочное или управляющее приложение может запустить службу с помощью функции StartService. Поскольку некоторые

приложения, основанные на работе служб, также должны в процессе загрузки

системы инициализировать свою работу, нет ничего необычного в том, что программа установки регистрирует службу в качестве автозапускаемой, запрашивая у пользователя перезагрузку системы для завершения установки службы и разрешения SCM запустить службу в ходе загрузки системы.

Когда программа вызывает функцию CreateService, она должна указать ряд

параметров, описывающих характеристики службы. Эти характеристики включают в себя:

- тип службы (служба относится к запускаемым в собственном процессе, а не

к тем, которые делят процесс с другими службами);

- размещение файла исполняемого образа службы;

- необязательное отображаемое имя службы и пароль, используемый для за-

пуска службы в конкретном контексте безопасности учетной записи;

- тип запуска, который показывает, должна ли служба автоматически запускаться в ходе загрузки системы или она должна запускаться произвольно по указанию SCP;

- код ошибки, показывающий, как система должна реагировать при обнаружении службой ошибки в процессе запуска;

SCM сохраняет каждую характеристику в виде параметра в разделе реестра,

относящемся к службе.

Типы данных в реестре:

REG_BINARY – двоичное число (по форме представления – 16ичное)
REG_DWORD – 4-ех байтовое целое число
REG_EXPAND_SZ – строковое значение переменной длины
REG_MULTI_SZ – список строковых значений
REG_SZ – строка фиксированной длины
REG_RESOURCE_LIST – список устройств в 16-ичной форме представления
REG_RESOURCE_REQUIREMENTS_LIST – список устройств, обсл. драйвером

REG_NONE – данные без определенного типа (часто записываются в реестр прикладными программами)
REG_LINK – строка в формате UNICODE
REG_QWORD – 64-битовое целое число

СТРУКТУРА РЕЕСТРА

Узел HKey_Classes_Root содержит типы файлов и связанные с ними приложения, например, .bmp связывается с программой отображения рисунков (Paint), doc – с программой Word и т.д.

HKey_Users хранит настройки (установки) всех пользователей системы.

Чтение и запись информации из реестра

```
#include "stdafx.h"

#include <windows.h>
#include <iostream>

using namespace std;

#define WIN_32_LEAN_AND_MEAN

void writeToRegistry(void)
{
  DWORD lRv;
  HKEY hKey;

  //Check if the registry exists

  lRv = RegOpenKeyEx(
    HKEY_CURRENT_USER,
    L"Software\\OV_German",
    0,
    KEY_WRITE,
    &hKey
  );

  if (lRv != ERROR_SUCCESS)
  {
    DWORD dwDisposition;

    // Create a key if it did not exist

    lRv = RegCreateKeyEx(
      HKEY_CURRENT_USER,
      L"Software\\OV_German",                  //"Use Multi-Byte Character Set" by using
L
      0,
      NULL,
      REG_OPTION_NON_VOLATILE,
      KEY_ALL_ACCESS,
      NULL,
      &hKey,
       &dwDisposition
    );

    DWORD dwValue = 14;


    RegSetValueEx(
      hKey,
      L"Read more books",
      0,
      REG_DWORD,
      reinterpret_cast<BYTE *>(&dwValue),
      sizeof(dwValue)
    );
```

```
    RegCloseKey(hKey);
  }
}

//Чтение из реестра
void readValueFromRegistry(void)
{


  HKEY hKey;

  //Check if the registry exists
  DWORD lRv = RegOpenKeyEx(
    HKEY_CURRENT_USER,
    L"Software\\OV_GERMAN",
    0,
    KEY_READ,
    &hKey
  );

  if (lRv == ERROR_SUCCESS)
  {
    DWORD BufferSize = sizeof(DWORD);
    DWORD dwRet;
    DWORD cbData;
    DWORD cbVal = 0;

    dwRet = RegQueryValueEx(
      hKey,
      L"Read more books",
      NULL,
      NULL,
      (LPBYTE)&cbVal,
      &cbData
    );

    if( dwRet == ERROR_SUCCESS )
      cout<<"\nValue of OV_German is " << cbVal << endl;
    else cout<<"\nRegQueryValueEx failed " << dwRet << endl;
  }
  else
  {
    cout<<"RegOpenKeyEx failed " << lRv << endl;
  }
}

int main()
{
  writeToRegistry();
  readValueFromRegistry();
  getchar();
  return 0;
}
```

++

Основные методы:

int CreateRegistryKey(CString ConstantKeyPath);

```
int ReadRegistryKeyAttributes(CString ConstantKeyPath);
int UpdateRegistryKeyAttribute(CString ConstantKeyPath, CString Attribute, DW
ORD value);
int DeleteRegistryKey(CString ConstantKeyPath);
```

Удаление узла из реестра

```
int DeleteRegistryKey(CString ConstantKeyPath)
{
	int lReturn = RegDeleteKey(HKEY_CURRENT_USER, ConstantKeyPath);
	return lReturn;
}
```

Создание узла реестра:

```
.....

	int iResult;
	CString KeyPath = _T("Software\\RCD_Technologies\\Rajib_Test");

	iResult =  CreateRegistryKey(KeyPath);
	if (iResult<0)
	{
		cout << "CreateRegistryKey Operation Failed" << endl;
		return -1;
	}
	cout << "<-- CreateRegistryKey Operation Successfull -->" << endl;
```

Открытие узла:

```
	HKEY MyRegistryKey;
	if (RegOpenKeyEx(HKEY_CURRENT_USER, ConstantKeyPath, 0, KEY_
ALL_ACCESS, &MyRegistryKey) != ERROR_SUCCESS)
	{
		cout << "KeyOpen Failed" << endl;
		return -1;
	}
```

Можно записать несколько значений:

```
lRv = RegCreateKeyEx(
	HKEY_CURRENT_USER,
	L"Software\\OV_German2",                //"Use Multi-Byte Character Set" by using
L
	0,
	NULL,
	REG_OPTION_NON_VOLATILE,
	KEY_ALL_ACCESS,
```

```
      NULL,
      &hKey,
       &dwDisposition
    );

    DWORD dwValue = 14;
       TCHAR  sr[]=TEXT("be happy");

    RegSetValueEx(
      hKey,
      L"Read more books",
      0,
      REG_DWORD,
      reinterpret_cast<BYTE *>(&dwValue),
      sizeof(dwValue)
    );

         RegSetValueEx(
      hKey,
      L"LG",
      0,
      REG_SZ,
      reinterpret_cast<BYTE *>(&sr), // строка не должна содержать пробелов!
      sizeof(dwValue)
    );



    RegCloseKey(hKey);




//Чтение из реестра
void readValueFromRegistry(void)
{


  HKEY hKey;

  //Check if the registry exists
  DWORD lRv = RegOpenKeyEx(
    HKEY_CURRENT_USER,
    L"Software\\OV_GERMAN",
    0,
    KEY_READ,
    &hKey
  );

  if (lRv == ERROR_SUCCESS)
  {
    DWORD BufferSize = sizeof(DWORD);
    DWORD dwRet;
    DWORD cbData;
    DWORD cbVal = 0;

    dwRet = RegQueryValueEx(
      hKey,
      L"Read more books",
      NULL,
      NULL,
      (LPBYTE)&cbVal,
      &cbData
    );
```

```
    if( dwRet == ERROR_SUCCESS )
      cout<<"\nValue of OV_German is " << cbVal << endl;
    else cout<<"\nRegQueryValueEx failed " << dwRet << endl;
  }
  else
  {
    cout<<"RegOpenKeyEx failed " << lRv << endl;
  }
}

int main()
{
  writeToRegistry();
  readValueFromRegistry();
  getchar();
  return 0;
}
```

Запись значения в реестр выполняется таким образом

```
RegSetValueEx(
      hKey,
      L"Read more books",
      0,
      REG_DWORD,
      reinterpret_cast<BYTE *>(&dwValue),
      sizeof(dwValue)
    );
```

Здесь указано, что атрибут имеет имя Read more books. Это целое число (`REG_DWORD`), ему присваивается значение переменной `dwValue`.

++

Сохранение ключа реестра в файле

```
//  Save registry key SubKey to file OutFile
//  Return TRUE if success, otherwise it returns FALSE
//  Note that the error code d is used only as a local variable information
BOOL SaveRegKeyPath(CString &Root, CString &SubKey, CString &OutFile)
{
  BOOL  ret=TRUE;
  HKEY  hKey=NULL;
  DWORD  d;
  HKEY  hRoot;

  //  Set SE_BACKUP_NAME privilege

  SetPrivilege(SE_BACKUP_NAME,TRUE);
  //  Determine the hive

  hRoot=(Root.CompareNoCase(HKCU)==0)?HKEY_CURRENT_USER:HKEY_LOCAL_MACHINE;
  //  We have to save only existing key ! (KEY_READ parameter below)
  if (RegOpenKeyEx(hRoot, SubKey,0,KEY_READ, &hKey)==ERROR_SUCCESS)  {
    if (IsFileExist(OutFile)){
```

```
      //we must delete file before saving, otherwise it doesn't work !
      if (DeleteFile(OutFile))
        d=RegSaveKey(hKey,OutFile,NULL);
    } else d=RegSaveKey(hKey,OutFile,NULL);
    if (d!=ERROR_SUCCESS)
   if (d!=ERROR_SUCCESS)
      ret=FALSE;

    RegCloseKey(hKey);
  }
  else {
    if (IsKeyExist(hRoot,SubKey)==FALSE)
      d=ERROR_FILE_NOT_FOUND;
    else d=0;
    ret=FALSE;
  }

  SetPrivilege(SE_BACKUP_NAME,FALSE);

  return ret;
}
```

++

Чтение ключа реестра из файла

```
// Restore registry key SubKey from file InFile
// If Force=TRUE then we force the restore operation
// Return TRUE if success, otherwise it returns FALSE
BOOL LoadRegKeyPath(CString &Root, CString &SubKey, CString &InFile, BOOL Force)
{
    BOOL    ret=TRUE;
    HKEY    hKey=NULL;
    DWORD    d;
    HKEY    hRoot;

    SetPrivilege(SE_RESTORE_NAME,TRUE);
    SetPrivilege(SE_BACKUP_NAME,TRUE);

    hRoot=(Root.CompareNoCase(HKCU)==0)?HKEY_CURRENT_USER:HKEY_LOCAL_MACHINE;
    if (!IsFileExist(InFile)) {
        d=ERROR_FILE_NOT_FOUND;
        PrintError(d);
        ret=FALSE;
    }
  else {
        HKEY    hhKey;
        char    lpClass[80];
        DWORD    lpDisposition=0;
        if (RegCreateKeyEx(hRoot,SubKey,0,lpClass, REG_OPTION_BACKUP_RESTORE,
            KEY_ALL_ACCESS, NULL, &hhKey,
            &lpDisposition)==ERROR_SUCCESS) {

            d=RegRestoreKey(hhKey,InFile,
              (Force==FALSE)?REG_NO_LAZY_FLUSH:REG_FORCE_RESTORE);
            RegCloseKey(hhKey);
            if (d!=ERROR_SUCCESS) {
                PrintError(d);
                ret=FALSE;
            }
```

```
        } else ret=FALSE;
    }

    SetPrivilege(SE_RESTORE_NAME,FALSE);
    SetPrivilege(SE_BACKUP_NAME,FALSE);

    return ret;
}
```