在 Windows 2000 上，安装实用工具通常会检查注册表的当前和最大大小，以确定是否有足够的可用空间来容纳将要添加的新数据。 此示例演示如何使用 System 对象中的“正在使用的注册表配额百分比”性能计数器以编程方式执行此操作。

以下示例使用性能数据帮助程序 (PDH) 来获取计数器值;它必须与 Pdh.lib 链接。 PDH 是一组高级 API，用于获取性能数据。

**备注**

无需在 Windows Server 2003 或 Windows XP 上实现此注册表大小检查，因为它们没有注册表配额限制。

C++复制

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

// Determines the current and maximum registry size.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <windows.h>

#include <tchar.h>

#include <stdio.h>

#include <pdh.h>

PDH\_STATUS GetRegistrySize( LPTSTR szMachineName,

LPDWORD lpdwCurrentSize, LPDWORD lpdwMaximumSize );

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

// Entry point for the program. This function demonstrates how to

// use the GetRegistrySize function implemented below.

//

// It will use the first argument, if present, as the name of the

// computer whose registry you wish to determine. If unspecified,

// it will use the local computer.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int \_tmain( int argc, TCHAR \*argv[] )

{

LPTSTR szMachineName = NULL;

PDH\_STATUS pdhStatus = 0;

DWORD dwCurrent = 0;

DWORD dwMaximum = 0;

// Allow a computer name to be specified on the command line.

if ( argc > 1 )

szMachineName = argv[1];

// Get the registry size.

pdhStatus=GetRegistrySize(szMachineName, &dwCurrent, &dwMaximum);

// Print the results.

if ( pdhStatus == ERROR\_SUCCESS )

{

\_tprintf( TEXT("Registry size: %ld bytes\n"), dwCurrent );

\_tprintf( TEXT("Max registry size: %ld bytes\n"), dwMaximum );

}

else

{

// If the operation failed, print the PDH error message.

LPTSTR szMessage = NULL;

FormatMessage( FORMAT\_MESSAGE\_ALLOCATE\_BUFFER |

FORMAT\_MESSAGE\_FROM\_HMODULE,

GetModuleHandle( TEXT("PDH.DLL") ), pdhStatus,

MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT),

szMessage, 0, NULL );

\_tprintf( TEXT("GetRegistrySize failed: %s"), szMessage );

LocalFree( szMessage );

}

return 0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

// Retrieves the current and maximum registry size. It gets this

// information from the raw counter values for the "% Registry Quota

// In Use" performance counter within the System object.

//

// PARAMETERS:

// szMachineName - Null-terminated string that specifies the

// name of the computer whose registry you wish to query.

// If this parameter is NULL, the local computer is used.

//

// lpdwCurrentSize - Receives the current registry size.

//

// lpdwMaximumSize - Receives the maximum registry size.

//

// RETURN VALUE:

// ERROR\_SUCCESS if successful. Otherwise, the function

// returns a PDH error code. These error codes can be

// found in PDHMSG.H. A textual error message can be

// retrieved from PDH.DLL using the FormatMessage function.

//

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

PDH\_STATUS GetRegistrySize( LPTSTR szMachineName,

LPDWORD lpdwCurrentSize, LPDWORD lpdwMaximumSize )

{

PDH\_STATUS pdhResult = 0;

TCHAR szCounterPath[1024];

DWORD dwPathSize = 1024;

PDH\_COUNTER\_PATH\_ELEMENTS pe;

PDH\_RAW\_COUNTER pdhRawValues;

HQUERY hQuery = NULL;

HCOUNTER hCounter = NULL;

DWORD dwType = 0;

// Open PDH query

pdhResult = PdhOpenQuery( NULL, 0, &hQuery );

if ( pdhResult != ERROR\_SUCCESS )

return pdhResult;

\_\_try

{

// Create counter path

pe.szMachineName = szMachineName;

pe.szObjectName = TEXT("System");

pe.szInstanceName = NULL;

pe.szParentInstance = NULL;

pe.dwInstanceIndex = 1;

pe.szCounterName = TEXT("% Registry Quota In Use");

pdhResult = PdhMakeCounterPath( &pe, szCounterPath,

&dwPathSize, 0 );

if ( pdhResult != ERROR\_SUCCESS )

\_\_leave;

// Add the counter to the query

pdhResult=PdhAddCounter(hQuery, szCounterPath, 0, &hCounter);

if ( pdhResult != ERROR\_SUCCESS )

\_\_leave;

// Run the query to collect the performance data

pdhResult = PdhCollectQueryData( hQuery );

if ( pdhResult != ERROR\_SUCCESS )

\_\_leave;

// Retrieve the raw counter data:

// The dividend (FirstValue) is the current registry size

// The divisor (SecondValue) is the maximum registry size

ZeroMemory( &pdhRawValues, sizeof(pdhRawValues) );

pdhResult = PdhGetRawCounterValue( hCounter, &dwType,

&pdhRawValues );

if ( pdhResult != ERROR\_SUCCESS )

\_\_leave;

// Store the sizes in variables.

if ( lpdwCurrentSize )

\*lpdwCurrentSize = (DWORD) pdhRawValues.FirstValue;

if ( lpdwMaximumSize )

\*lpdwMaximumSize = (DWORD) pdhRawValues.SecondValue;

}

\_\_finally

{

// Close the query

PdhCloseQuery( hQuery );

}

return 0;

}