# 这一节我们来学习如何遍历磁盘卷

# 学习大纲

|  |
| --- |
|  |

# 所谓的卷，就是指磁盘分区，一个卷就是一个磁盘分区

# API说明

## 1. GetLogicalDriveStrings

|  |
| --- |
|  |

## 演练，

## 1.创建一个c++常规空项目，取名：lesson12-travel-disk-volume，然后新建一个cpp源文件，先学习GetLogicalDriveStrings，有2个代码实例

## 2.实例1

### travel-volume.cpp

|  |
| --- |
| #include<Windows.h>  #include<stdio.h>  #include<stdlib.h>  #define BUF\_SIZE 1024  //注意：这里有坑，使用vs2008编程，最好先把所有变量在上面声明，然后再在下面赋值，否则结果不对  int main()  {  CHAR driveStrs[BUF\_SIZE];  PCHAR pDrive;  ZeroMemory(driveStrs,BUF\_SIZE);  DWORD dwSize = GetLogicalDriveStrings(BUF\_SIZE-1,driveStrs);  if(!dwSize)  {  MessageBox(NULL,"get drive strings failed","error",0);  }    //printf("drive strings: %s\n",driveStrs);  pDrive = (PCHAR)driveStrs;  printf("logical drives:\n");  do  {  printf("%s\n",pDrive);  pDrive +=(lstrlen(pDrive)+1);  }while(\*pDrive!='\x00');  system("pause");  return 0;  } |

### 效果：

|  |
| --- |
|  |

## 3.实例2.

### travel-volume.cpp

|  |
| --- |
| #include <Windows.h>  #include <stdio.h>  #include <stdlib.h>  #define BUFSIZE 1024  BOOL GetDriverInfo(LPSTR szDrive);  //int WinMain(HINSTANCE hInstance,HINSTANCE hPreInstance,LPSTR lpCmdLine,int nCmdShow)  int main(void)  {  CHAR szLogicDriveStrings[BUFSIZE];  PCHAR szDrive;  ZeroMemory(szLogicDriveStrings,BUFSIZE);  GetLogicalDriveStrings(BUFSIZE-1,szLogicDriveStrings);  szDrive = (PCHAR)szLogicDriveStrings;  do  {  if(!GetDriverInfo(szDrive))  {  printf("\nGet Volume Information Error:%d",GetLastError());  }  szDrive += (lstrlen(szDrive)+1);  } while (\*szDrive !='\x00');  system("PAUSE");  return 0;  }  BOOL GetDriverInfo(LPSTR szDrive)  {  UINT uDriverType;  DWORD dwVolumeSerialNumber;  DWORD dwMaximumComponentlength;  DWORD dwFileSystemFlags;  CHAR szFileSystemNameBuffer[BUFSIZE];  CHAR szDriveName[MAX\_PATH];  printf("\n%s\n",szDrive);  uDriverType = GetDriveType(szDrive);  switch(uDriverType)  {  case DRIVE\_UNKNOWN:  printf("The driver type cannot be determined!");  break;  case DRIVE\_NO\_ROOT\_DIR:  printf("The root path is invalid,for example,no volume is mounted at the path");  break;  case DRIVE\_REMOVABLE:  printf("The drive is a type that has removable media,for example:a floppy drive or removable hard disk");  break;  case DRIVE\_FIXED:  printf("The drive is a type that cannot be removed, for example,a fixed hard drive");  break;  case DRIVE\_REMOTE:  printf("This drive is a remote(network) drive");  break;  case DRIVE\_CDROM:  printf("This drive is a CD-ROM drive.");  break;  case DRIVE\_RAMDISK:  printf("This drive is a RAM disk");  break;  default:  break;  }  if (!(GetVolumeInformation(  szDrive,  szDriveName,  MAX\_PATH,  &dwVolumeSerialNumber,  &dwMaximumComponentlength,  &dwFileSystemFlags,  szFileSystemNameBuffer,  BUFSIZE)))  {  return FALSE;  }  if (0!=lstrlen(szDriveName))  {  printf("\nDrive Name is %s.\n",szDriveName);  }  printf("\nVolume Serial is %u.",dwVolumeSerialNumber );  printf("\nMaximum Component Length is %u.",dwMaximumComponentlength);  printf("\nSystem Type is %s.\n",szFileSystemNameBuffer);    if (dwFileSystemFlags & FILE\_VOLUME\_QUOTAS)  {  printf("The file system supports disk Quotas.\n");  }  if (dwFileSystemFlags & FILE\_SUPPORTS\_REPARSE\_POINTS)  {  printf("The file system does not support volume mount points.\n");  }  if (dwFileSystemFlags & FILE\_CASE\_SENSITIVE\_SEARCH)  {  printf("The file system supports case-sentitive file name.\n");  }  printf("...\n");  return TRUE;  } |

## 效果：

|  |
| --- |
| C:\  The drive is a type that cannot be removed, for example,a fixed hard drive  Volume Serial is 3363692264.  Maximum Component Length is 255.  System Type is NTFS.  The file system supports disk Quotas.  The file system does not support volume mount points.  The file system supports case-sentitive file name.  ...  D:\  The drive is a type that cannot be removed, for example,a fixed hard drive  Drive Name is 数据盘.  Volume Serial is 1550604159.  Maximum Component Length is 255.  System Type is NTFS.  The file system supports disk Quotas.  The file system does not support volume mount points.  The file system supports case-sentitive file name.  ...  E:\  The drive is a type that cannot be removed, for example,a fixed hard drive  Drive Name is 新加卷.  Volume Serial is 542590885.  Maximum Component Length is 255.  System Type is NTFS.  The file system supports disk Quotas.  The file system does not support volume mount points.  The file system supports case-sentitive file name.  ...  F:\  The drive is a type that cannot be removed, for example,a fixed hard drive  Drive Name is Document and software.  Volume Serial is 3639735162.  Maximum Component Length is 255.  System Type is NTFS.  The file system supports disk Quotas.  The file system does not support volume mount points.  The file system supports case-sentitive file name.  ...  G:\  The drive is a type that cannot be removed, for example,a fixed hard drive  Drive Name is learning.  Volume Serial is 3629183472.  Maximum Component Length is 255.  System Type is NTFS.  The file system supports disk Quotas.  The file system does not support volume mount points.  The file system supports case-sentitive file name. |

## 演练2.FindFirstVolume和FindNextVolume还有FindVolumeClose

## 1.新建一个常规空项目，取名Lesson12\_2-findVolume，添加一个cpp文件，添加一些代码

|  |
| --- |
| #include<Windows.h>  #include<stdio.h>  #include<stdlib.h>  int main()  {  HANDLE hVol;  CHAR szVolume[MAX\_PATH] = { 0 };  hVol = FindFirstVolumeA(szVolume,MAX\_PATH);  if(hVol == INVALID\_HANDLE\_VALUE)  {  MessageBox(NULL,"Find First Volume Failed...","error",0);  }  printf("First Volume:%s\n",szVolume);  FindVolumeClose(hVol);  system("pause");  return 0;  } |

### 编译运行程序，效果如下

|  |
| --- |
|  |

### 得到的结果和我预期的不一样。那这么处理这种数据呢？这是一个设备名称，这是window的GUID标记。

## 2.我们可以利用循环调用FindNextVolume来找到下一个，代码如下

|  |
| --- |
| #include<Windows.h>  #include<stdio.h>  #include<stdlib.h>  int main()  {  HANDLE hVol;  CHAR szVolume[MAX\_PATH] = { 0 };  hVol = FindFirstVolumeA(szVolume,MAX\_PATH);  BOOL bFlag;  if(hVol == INVALID\_HANDLE\_VALUE)  {  MessageBox(NULL,"Find First Volume Failed...","error",0);  }  printf("First Volume:%s\n",szVolume);  while(FindNextVolume(hVol,szVolume,MAX\_PATH))  {  printf("NextVolume:%s\n",szVolume);  }  bFlag = FindVolumeClose(hVol);  system("pause");  return bFlag;  } |

### 效果：

|  |
| --- |
|  |

### 那么问题来了，这些数据这么使用？只有在window高级编程的时候使用