在 .NET Framework 中，可以使用以下类来访问文件系统信息：

* [System.IO.FileInfo](https://msdn.microsoft.com/zh-cn/library/system.io.fileinfo.aspx)
* [System.IO.DirectoryInfo](https://msdn.microsoft.com/zh-cn/library/system.io.directoryinfo.aspx)
* [System.IO.DriveInfo](https://msdn.microsoft.com/zh-cn/library/system.io.driveinfo.aspx)
* [System.IO.Directory](https://msdn.microsoft.com/zh-cn/library/system.io.directory.aspx)
* [System.IO.File](https://msdn.microsoft.com/zh-cn/library/system.io.file.aspx)

[FileInfo](https://msdn.microsoft.com/zh-cn/library/system.io.fileinfo.aspx) 和 [DirectoryInfo](https://msdn.microsoft.com/zh-cn/library/system.io.directoryinfo.aspx) 类表示文件或目录，包含公开 NTFS 文件系统所支持的很多文件特性的属性，同时还包含用于打开、关闭、移动和删除文件和文件夹的方法。可以通过将表示文件、文件夹或驱动器名称的字符串传递到下面的构造函数来创建这些类的实例：

System.IO.DriveInfo di = new System.IO.DriveInfo(@"C:\");

此外，还可以通过调用 [DirectoryInfo.GetDirectories](https://msdn.microsoft.com/zh-cn/library/s7xk2b58.aspx)、[DirectoryInfo.GetFiles](https://msdn.microsoft.com/zh-cn/library/4cyf24ss.aspx) 和 [DriveInfo.RootDirectory](https://msdn.microsoft.com/zh-cn/library/system.io.driveinfo.rootdirectory.aspx) 来获取文件、文件夹或驱动器的名称。

[System.IO.Directory](https://msdn.microsoft.com/zh-cn/library/system.io.directory.aspx) 和 [System.IO.File](https://msdn.microsoft.com/zh-cn/library/system.io.file.aspx) 类提供用于检索有关目录和文件的信息的静态方法。

下面的示例演示访问有关文件和文件夹的信息的各种方法。

|  |
| --- |
| class FileSysInfo  {  static void Main()  {  // You can also use System.Environment.GetLogicalDrives to  // obtain names of all logical drives on the computer.  System.IO.DriveInfo di = new System.IO.DriveInfo(@"C:\");  Console.WriteLine(di.TotalFreeSpace);  Console.WriteLine(di.VolumeLabel);  // Get the root directory and print out some information about it.  System.IO.DirectoryInfo dirInfo = di.RootDirectory;  Console.WriteLine(dirInfo.Attributes.ToString());  // Get the files in the directory and print out some information about them.  System.IO.FileInfo[] fileNames = dirInfo.GetFiles("\*.\*");  foreach (System.IO.FileInfo fi in fileNames)  {  Console.WriteLine("{0}: {1}: {2}", fi.Name, fi.LastAccessTime, fi.Length);  }  // Get the subdirectories directly that is under the root.  // See "How to: Iterate Through a Directory Tree" for an example of how to  // iterate through an entire tree.  System.IO.DirectoryInfo[] dirInfos = dirInfo.GetDirectories("\*.\*");  foreach (System.IO.DirectoryInfo d in dirInfos)  {  Console.WriteLine(d.Name);  }  // The Directory and File classes provide several static methods  // for accessing files and directories.  // Get the current application directory.  string currentDirName = System.IO.Directory.GetCurrentDirectory();  Console.WriteLine(currentDirName);  // Get an array of file names as strings rather than FileInfo objects.  // Use this method when storage space is an issue, and when you might  // hold on to the file name reference for a while before you try to access  // the file.  string[] files = System.IO.Directory.GetFiles(currentDirName, "\*.txt");  foreach (string s in files)  {  // Create the FileInfo object only when needed to ensure  // the information is as current as possible.  System.IO.FileInfo fi = null;  try  {  fi = new System.IO.FileInfo(s);  }  catch (System.IO.FileNotFoundException e)  {  // To inform the user and continue is  // sufficient for this demonstration.  // Your application may require different behavior.  Console.WriteLine(e.Message);  continue;  }  Console.WriteLine("{0} : {1}",fi.Name, fi.Directory);  }  // Change the directory. In this case, first check to see  // whether it already exists, and create it if it does not.  // If this is not appropriate for your application, you can  // handle the System.IO.IOException that will be raised if the  // directory cannot be found.  if (!System.IO.Directory.Exists(@"C:\Users\Public\TestFolder\"))  {  System.IO.Directory.CreateDirectory(@"C:\Users\Public\TestFolder\");  }  System.IO.Directory.SetCurrentDirectory(@"C:\Users\Public\TestFolder\");  currentDirName = System.IO.Directory.GetCurrentDirectory();  Console.WriteLine(currentDirName);  // Keep the console window open in debug mode.  Console.WriteLine("Press any key to exit.");  Console.ReadKey();  }  } |

## 我的小例子

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.IO;  namespace Drive\_dir\_file\_info  {  internal class Program  {  static void GetInfos()  {  DriveInfo dri = new DriveInfo(@"e:\");  DirectoryInfo di = dri.RootDirectory;  string currDir = di.FullName;  Console.WriteLine("当前路径: "+currDir);  string[] files = Directory.GetFiles(currDir);//获取到的只是文件没有目录  foreach (string file in files)  {  Console.WriteLine(file);  }  //string[] dirs = Directory.GetDirectories(currDir);//获取文件夹  //foreach (string dir in dirs)  //{  // Console.WriteLine(dir);  //}  DirectoryInfo[] dirInfos = di.GetDirectories(@"123");//注意这里是二级目录不能有驱动器号,否则报错  foreach (var info in dirInfos)  {  FileInfo[] fileNames = info.GetFiles("\*.\*");  foreach (FileInfo fi in fileNames)  {  Console.WriteLine("{0}:{1}",fi.Name,fi.Length);  }    }  }  static void Main(string[] args)  {  GetInfos();  }  }  } |

**可靠编程**

处理用户指定的路径字符串时，还应处理在以下情况下引发的异常：

* 文件名的格式不正确。例如，包含无效字符或仅包含空白。
* 文件名为空。
* 文件名长于系统定义的最大长度。
* 文件名包含冒号 (:)。

如果应用程序不具有读取指定文件所需的足够权限，则无论路径是否存在，**Exists** 方法都将返回 **false**；该方法不引发异常。

**请参阅**

[System.IO](https://msdn.microsoft.com/zh-cn/library/system.io.aspx)

[C# 编程指南](https://msdn.microsoft.com/zh-cn/library/67ef8sbd.aspx)

[文件系统和注册表（C# 编程指南）](https://msdn.microsoft.com/zh-cn/library/2kzb96fk.aspx)

# 总结:

## 1,创建DriveInfo实例对象

### System.IO.DriveInfo di = new System.IO.DriveInfo(@"C:\");

## 2.获取DirectoryInfo实例对象

### System.IO.DirectoryInfo dirInfo = di.RootDirectory;

## 3.获取当前路径

### string currentDirName = System.IO.Directory.GetCurrentDirectory();

## 4.获取当前路径下面的所有txt文件

### string[] files = System.IO.Directory.GetFiles(currentDirName, "\*.txt");

## 5.创建FileInfo实例对象

### FileInfo fi = new System.IO.FileInfo(s);

## 6.判断一个路径是否存在

### System.IO.Directory.Exists(@"C:\Users\Public\TestFolder\")

## 7.创建新文件夹

### System.IO.Directory.CreateDirectory(@"C:\Users\Public\TestFolder\");

## 8.设置当前路径

### System.IO.Directory.SetCurrentDirectory(@"C:\Users\Public\TestFolder\");

### 注意:

### 1.在c#中获取文件和获取目录的分开的,不像win32APi和文件存在那样子把文件和目录混在一起

#### 2. Directory和DirectoryInfo都有GetFiles方法但是他们的返回值不一样, :Directory类的GetFiles方法方法返回一个字符串数组, DirectoryInfo类的GetFiles方法返回FileInfo对象数组

#### 3. Directory和DirectoryInfo都有GetDirectories方法但是他们的返回值不一样, :Directory类的GetDirectories方法返回一个字符串数组,DirectoryInfo类的GetDirectories方法返回DirectoryInfo对象数组