File.Copy Method

Reference

Definition

Namespace: System.IO

Assembly: System.Runtime.dll

Copies an existing file to a new file.

Overloads

Expand table

Copy(String, String)	Copies an existing file to a new file. Overwriting a file of the same name is not allowed.
Copy(String, String, Boolean)	Copies an existing file to a new file. Overwriting a file of the same name is allowed.

Copy(String, String)

Source: File.cs

Copies an existing file to a new file. Overwriting a file of the same name is not allowed.

C#

```
public static void Copy (string sourceFileName, string
destFileName);
```

Parameters

sourceFileName String

The file to copy.

destFileName String

The name of the destination file. This cannot be a directory or an existing file.

Exceptions

UnauthorizedAccessException

The caller does not have the required permission.

ArgumentException

`sourceFileName` or `destFileName` is a zero-length string, contains only white space, or contains one or more invalid characters. You can query for invalid characters by using the GetInvalidPathChars() method.

-or-

`sourceFileName` or `destFileName` specifies a directory.

ArgumentNullException

`sourceFileName` or `destFileName` is `null`.

PathTooLongException

The specified path, file name, or both exceed the system-defined maximum length.

DirectoryNotFoundException

The path specified in `sourceFileName` or `destFileName` is invalid (for example, it is on an unmapped drive).

FileNotFoundException

`sourceFileName` was not found.

IOException

`destFileName` exists.

-or-

An I/O error has occurred.

NotSupportedException

`sourceFileName` or `destFileName` is in an invalid format.

Examples

The following example copies files to the C:\archives\2008 backup folder. It uses the two overloads of the Copy method as follows:

- It first uses the File.Copy(String, String) method overload to copy text (.txt) files. The code demonstrates that this overload does not allow overwriting files that were already copied.
- It then uses the File.Copy(String, String, Boolean) method overload to copy pictures (.jpg files). The code demonstrates that this overload does allow overwriting files that were already copied.

C#

```
string sourceDir = @"c:\current";
string backupDir = @"c:\archives\2008";

try
{
    string[] picList = Directory.GetFiles(sourceDir, "*.jpg");
    string[] txtList = Directory.GetFiles(sourceDir, "*.txt");

    // Copy picture files.
    foreach (string f in picList)
    {
        // Remove path from the file name.
        string fName = f.Substring(sourceDir.Length + 1);

        // Use the Path.Combine method to safely append the file
name to the path.
        // Will overwrite if the destination file already exists.
        File.Copy(Path.Combine(sourceDir, fName),
Path.Combine(backupDir, fName), true);
    }

    // Copy text files.
    foreach (string f in txtList)
    {

        // Remove path from the file name.
        string fName = f.Substring(sourceDir.Length + 1);

        try
        {
            // Will not overwrite if the destination file already
exists.
            File.Copy(Path.Combine(sourceDir, fName),
Path.Combine(backupDir, fName));
        }

        // Catch exception if the file was already copied.
        catch (IOException copyError)
        {
            Console.WriteLine(copyError.Message);
        }
    }
```

```
    // Delete source files that were copied.
    foreach (string f in txtList)
    {
        File.Delete(f);
    }
    foreach (string f in picList)
    {
        File.Delete(f);
    }
}

catch (DirectoryNotFoundException dirNotFound)
{
    Console.WriteLine(dirNotFound.Message);
}
```

Remarks

This method is equivalent to the Copy(String, String, Boolean) method overload with the `overwrite` parameter set to `false`.

The `sourceFileName` and `destFileName` parameters can specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. To obtain the current working directory, see the Directory.GetCurrentDirectory method. This method does not support wildcard characters in the parameters.

The attributes of the original file are retained in the copied file.

See also

- Move(String, String)
- Move(String, String)
- File and Stream I/O
- Reading Text From A File
- How to: Write Text to a File
- How to: Read and Write to a Newly Created Data File

Applies to

▼ .NET 9 and other versions

Product	Versions
.NET	Core 1.0, Core 1.1, Core 2.0, Core 2.1, Core 2.2, Core 3.0, Core 3.1, 5, 6, 7, 8, 9
.NET Framework	1.1, 2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1
.NET Standard	1.3, 1.4, 1.6, 2.0, 2.1
UWP	10.0

Copy(String, String, Boolean)

Source: File.cs

Copies an existing file to a new file. Overwriting a file of the same name is allowed.

C#

```
public static void Copy (string sourceFileName, string destFile-
Name, bool overwrite);
```

Parameters

`sourceFileName` String

The file to copy.

`destFileName` String

The name of the destination file. This cannot be a directory.

`overwrite` Boolean

`true` if the destination file should be replaced if it already exists; otherwise, `false`.

Exceptions

UnauthorizedAccessException

The caller does not have the required permission.

-or-

`destFileName` is read-only.

-or-

`overwrite` is `true`, `destFileName` exists and is hidden, but `sourceFileName` is not hidden.

ArgumentException

`sourceFileName` or `destFileName` is a zero-length string, contains only white space, or contains one or more invalid characters. You can query for invalid characters by using the GetInvalidPathChars() method.

-or-

`sourceFileName` or `destFileName` specifies a directory.

ArgumentNullException

`sourceFileName` or `destFileName` is `null`.

PathTooLongException

The specified path, file name, or both exceed the system-defined maximum length.

DirectoryNotFoundException

The path specified in `sourceFileName` or `destFileName` is invalid (for example, it is on an unmapped drive).

FileNotFoundException

`sourceFileName` was not found.

IOException

`destFileName` exists and `overwrite` is `false`.

-or-

An I/O error has occurred.

NotSupportedException

`sourceFileName` or `destFileName` is in an invalid format.

Examples

The following example copies files to the C:\archives\2008 backup folder. It uses the two overloads of the Copy method as follows:

- It first uses the File.Copy(String, String) method overload to copy text (.txt) files. The code demonstrates that this overload does not allow overwriting files that

were already copied.

It then uses the File.Copy(String, String, Boolean) method overload to copy pictures (.jpg files). The code demonstrates that this overload does allow overwriting files that were already copied.

C#

```
string sourceDir = @"c:\current";
string backupDir = @"c:\archives\2008";

try
{
    string[] picList = Directory.GetFiles(sourceDir, "*.jpg");
    string[] txtList = Directory.GetFiles(sourceDir, "*.txt");

    // Copy picture files.
    foreach (string f in picList)
    {
        // Remove path from the file name.
        string fName = f.Substring(sourceDir.Length + 1);

        // Use the Path.Combine method to safely append the file
name to the path.
        // Will overwrite if the destination file already exists.
        File.Copy(Path.Combine(sourceDir, fName),
Path.Combine(backupDir, fName), true);
    }

    // Copy text files.
    foreach (string f in txtList)
    {

        // Remove path from the file name.
        string fName = f.Substring(sourceDir.Length + 1);

        try
        {
            // Will not overwrite if the destination file already
exists.
            File.Copy(Path.Combine(sourceDir, fName),
Path.Combine(backupDir, fName));
        }

        // Catch exception if the file was already copied.
        catch (IOException copyError)
        {
            Console.WriteLine(copyError.Message);
        }
    }

    // Delete source files that were copied.
    foreach (string f in txtList)
```

```
    {
        File.Delete(f);
    }
    foreach (string f in picList)
    {
        File.Delete(f);
    }
}

catch (DirectoryNotFoundException dirNotFound)
{
    Console.WriteLine(dirNotFound.Message);
}
```

Remarks

The `sourceFileName` and `destFileName` parameters can specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. This method does not support wildcard characters in the parameters.

The attributes of the original file are retained in the copied file.

For a list of common I/O tasks, see Common I/O Tasks.

See also

- Move(String, String)
- Move(String, String)
- File and Stream I/O
- Reading Text From A File
- How to: Write Text to a File
- How to: Read and Write to a Newly Created Data File

Applies to

▼ .NET 9 and other versions

Product	Versions
.NET	Core 1.0, Core 1.1, Core 2.0, Core 2.1, Core 2.2, Core 3.0, Core 3.1, 5, 6, 7, 8, 9
.NET Framework	1.1, 2.0, 3.0, 3.5, 4.0, 4.5, 4.5.1, 4.5.2, 4.6, 4.6.1, 4.6.2, 4.7, 4.7.1, 4.7.2, 4.8, 4.8.1

Product	Versions
.NET Standard	1.3, 1.4, 1.6, 2.0, 2.1
UWP	10.0

Collaborate with us on GitHub

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see our contributor guide.

.NET feedback

.NET is an open source project. Select a link to provide feedback:

Open a documentation issue

Provide product feedback