

System.IO.Directory Class

```
[ILASM]
.class public sealed Directory extends System.Object

[C#]
public sealed class Directory
```

Assembly Info:

- *Name:* mscorlib
- *Public Key:* [00 00 00 00 00 00 00 00 00 04 00 00 00 00 00 00]
- *Version:* 1.0.x.x
- *Attributes:*
 - CLSCompliantAttribute(true)

Summary

Provides information and performs operations on directories.

Inherits From: System.Object

Library: BCL

Thread Safety: All public static members of this type are safe for multithreaded operations. No instance members are guaranteed to be thread safe.

Description

Implementations are required to preserve the case of file and directory path strings, and to be case sensitive if and only if the current platform is case-sensitive.

[*Note:* In most **Directory** methods that accept *path* arguments, the path can refer to a file or a directory.]

Directory.Delete(System.String) Method

```
[ILASM]  
.method public hidebysig static void Delete(string path)  
  
[C#]  
public static void Delete(string path)
```

Summary

Deletes the empty directory specified in *path*.

Parameters

Parameter	Description
<i>path</i>	A System.String containing the name of the directory to delete. This directory must be writable and empty.

Description

This method behaves identically to **System.IO.Directory.Delete(*path*, false)**.

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory**.]

Exceptions

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.DirectoryNotFoundException	The specified <i>path</i> was not found.
System.IO.IOException	The directory specified by <i>path</i> is read-only or is not empty.
System.Security.SecurityException	The caller does not have the required permission.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to write to the specified directory. See System.Security.Permissions.FileIOPermissionAccess: Write .

5
6
7

1 Directory.Delete(System.String, 2 System.Boolean) Method

```
3 [ILASM]  
4 .method public hidebysig static void Delete(string path,  
5 bool recursive)  
  
6 [C#]  
7 public static void Delete(string path, bool recursive)
```

8 Summary

9 Deletes the specified directory and, if indicated, any subdirectories in
10 the directory.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the name of the directory to delete. This directory must be writable and cannot contain files unless <i>recursive</i> is true.
<i>recursive</i>	Specify true to delete subdirectories and files in <i>path</i> ; otherwise, specify false .

14
15

Description

16 The *path* argument is permitted to specify relative or absolute path
17 information. Relative path information is interpreted as relative to the
18 current working directory. [Note: To obtain the current working
19 directory, see **System.IO.Directory.GetCurrentDirectory**.]

20 Exceptions

21
22

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.DirectoryNotFoundException	The specified <i>path</i> was not found.
System.IO.IOException	The directory specified by <i>path</i> is read-only, or <i>recursive</i> is false and <i>path</i> is not an empty directory.
System.Security.SecurityException	The caller does not have the required

	permission.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to write to the specified directory. See System.Security.Permissions.FileIOPermissionAccess Write.

5
6
7

1 Directory.Exists(System.String) Method

```
2 [ILASM]  
3 .method public hidebysig static bool Exists(string path)  
4 [C#]  
5 public static bool Exists(string path)
```

6 Summary

7 Returns a **System.Boolean** indicating whether the specified directory
8 exists.

9 Parameters

10
11

Parameter	Description
<i>path</i>	A System.String containing the name of the directory to check.

12
13
14

13 Return Value

15 **true** if the caller has the required permissions and *path* contains the
16 name of an existing directory; otherwise, **false**. If *path* is **null**, a zero-
17 length string, or contains the name of a file, returns **false**.

18 Description

19 If the caller does not have sufficient permissions to read the files in the
20 directory specified by *path*, no exception is thrown and the method
21 returns **false** regardless of the existence of *path*.

22
23
24
25
26

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory**.]

27 Exceptions

28
29

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

System.IO.DirectoryNotFoundException *path* was not found.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to read the specified directory. See System.Security.Permissions.FileIOPermissionAccess . Read.

5
6
7

1 Directory.GetCreationTime(System.String)

2 Method

```
3 [ILASM]  
4 .method public hidebysig static valuetype System.DateTime  
5 GetCreationTime(string path)  
  
6 [C#]  
7 public static DateTime GetCreationTime(string path)
```

8 Summary

9 Returns the creation date and time of the specified file or directory.

10 Parameters

11
12

Parameter	Description
<i>path</i>	A System.String containing the name of the file or directory for which to obtain creation date and time information.

13
14
15

14 Return Value

16 A **System.DateTime** structure set to the creation date and time for
17 the specified directory. This value is expressed in local time.

18
19 Platforms that do not support this feature return
20 **System.DateTime.MinValue**.

21 Description

22 This method is equivalent to **System.IO.File.GetCreationTime**
23 (*path*).

24
25 The *path* argument is permitted to specify relative or absolute path
26 information. Relative path information is interpreted as relative to the
27 current working directory. [Note: To obtain the current working
28 directory, see **System.IO.Directory.GetCurrentDirectory**.]

29 Exceptions

30
31

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.

System.ArgumentNullException	<i>path</i> is null .
System.IO.IOException	The directory specified by <i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to read the specified file or directory See System.Security.Permissions.FileIOPermissionAccess.Read .

5
6
7

1 Directory.GetCurrentDirectory() Method

```
2 [ILASM]  
3 .method public hidebysig static string  
4 GetCurrentDirectory()  
  
5 [C#]  
6 public static string GetCurrentDirectory()
```

7 Summary

8 Returns the application's current working directory.

9 Return Value

10

11 A **System.String** containing the path of the current working directory.

12

13 Platforms that do not support this feature return
14 **System.String.Empty**.

15 Exceptions

16

17

Exception	Condition
System.Security.SecurityException	The caller does not have the required permission.

18

19 Permissions

20

21

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the current directory. See System.Security.Permissions.FileIOPermissionAccess.PathDiscovery

22

23

24

1 Directory.GetDirectories(System.String)

2 Method

```
3 [ILASM]  
4 .method public hidebysig static class System.String[]  
5 GetDirectories(string path)  
  
6 [C#]  
7 public static string[] GetDirectories(string path)
```

8 Summary

9 Returns the names of subdirectories in the specified directory.

10 Parameters

11
12

Parameter	Description
<i>path</i>	A System.String containing the name of the directory for which an array of subdirectory names is returned.

13
14
15

14 Return Value

16 A **System.String** array containing the names of subdirectories in
17 *path*.

18 Description

19 This method is identical to **System.IO.Directory.GetDirectories**
20 (*path*, "").

21
22
23
24
25
26
27
28

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory**.]

The names returned by this method are prefixed with the directory information provided in *path*.

29 Exceptions

30
31

Exception	Condition
System.ArgumentNullException	<i>path</i> is null .
System.Security.SecurityException	The caller does not have the required permission.

System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains implementation-defined invalid characters.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.IO.IOException	<i>path</i> is a file name.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory and its subdirectories. See System.Security.Permissions.FileIOPermissionAccess.PathDiscovery .

5
6
7

1 Directory.GetDirectories(System.String, 2 System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static class System.String[]  
5 GetDirectories(string path, string searchPattern)  
  
6 [C#]  
7 public static string[] GetDirectories(string path, string  
8 searchPattern)
```

9 Summary

10 Returns the names of subdirectories in the specified directory that
11 match the specified search pattern.

12 Parameters

13
14

Parameter	Description
<i>path</i>	A System.String containing the starting location for the search.
<i>searchPattern</i>	A System.String containing the text pattern to match against the names of subdirectories of <i>path</i> . <i>searchPattern</i> cannot contain System.IO.Path.DirectorySeparatorChar or System.IO.Path.AltDirectorySeparatorChar .

15
16
17

16 Return Value

18 A **String** array containing the names of subdirectories of *path* that
19 match *searchPattern*.

20 Description

21 The following wild card specifiers are permitted in *searchPattern*:

Wild card	Description
*	Zero or more characters.
?	Exactly one character.

22
23
24
25
26
27
28

The period (".") character, if immediately followed by a wild card specifier, indicates that the period or the empty string matches the pattern. For example, "foo.*" and "foo.?" match "foo". Note that "foo.*" and "foo*" behave identically. If the period is not immediately followed by a wildcard, it has no special meaning (it represents a period).

1
2
3
4
5
6
7
8
9
10

Characters other than the wild card specifiers represent themselves, for example, the *searchPattern* string "*" searches for all names in *path* ending with the letter "t". The *searchPattern* string "s*" searches for all names in *path* beginning with the letter "s".

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory.**]

11
12
13

Exceptions

Exception	Condition
System.ArgumentNullException	<i>path</i> or <i>searchPattern</i> is null .
System.Security.SecurityException	The caller does not have permission to access the requested information.
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains implementation-defined invalid characters. <i>searchPattern</i> does not contain a valid pattern.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.IO.IOException	<i>path</i> is a file name.

14
15
16
17

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory and its subdirectories. See System.Security.Permissions.FileIOPermissionAccess.PathDiscovery .

18
19
20

1 Directory.GetDirectoryRoot(System.String 2) Method

```
3 [ILASM]  
4 .method public hidebysig static string  
5 GetDirectoryRoot(string path)  
  
6 [C#]  
7 public static string GetDirectoryRoot(string path)
```

8 Summary

9 Returns the path root component of the specified path.

10 Parameters

11
12

Parameter	Description
<i>path</i>	A System.String containing the name of a file or directory.

13
14
15

Return Value

16 A **System.String** containing the root information for the specified
17 path.
18
19 Platforms that do not support this feature return
20 **System.String.Empty**.

21 Description

22 This method obtains the full path information for *path*, as returned by
23 **System.IO.Path.GetFullPath** (*path*) and returns the path root
24 component. The specified path is not required to exist.

25
26 The *path* argument is permitted to specify relative or absolute path
27 information. Relative path information is interpreted as relative to the
28 current working directory. [Note: To obtain the current working
29 directory, see **System.IO.Directory.GetCurrentDirectory**.]

30 Exceptions

31
32

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space. or contains one or more

	implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.Security.SecurityException	The caller does not have the required permission.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

1

2 **Example**

3

4 The following example demonstrates the
5 **System.IO.Directory.GetDirectoryRoot** method.

6
7

[C#]

```

8     using System;
9     using System.IO;
10    class GetDirectoryTest {
11        public static void Main() {
12            string [] paths = {
13
14                @"\\ecmatest\examples\pathtests.txt",
15                "pathtests.xyzy",
16                @"\",
17                @"C:\",
18                @"\\myserver\myshare\foo\bar\baz.txt"
19            };
20            foreach (string pathString in paths) {
21                string s = Directory.GetDirectoryRoot(pathString);
22                Console.WriteLine("Path: {0} Directory Root is
23 {1}",pathString, s== null? "null":s);
24            }
25        }
26    }

```

27 The output is

28

29 Path: \\ecmatest\examples\pathtests.txt Directory Root is
30 C:\

31

32

33 Path: pathtests.xyzy Directory Root is C:\

1
2
3
4
5
6
7
8
9
10
11

Path: \ Directory Root is C:\

Path: C:\ Directory Root is C:\

Path: \\myserver\myshare\foo\bar\baz.txt Directory Root is
\\myserver\myshare

12 **Permissions**
13
14

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified file or directory. See System.Security.Permissions.FileIOPermissionAccess: PathDiscovery

15
16
17

Directory.GetFiles(System.String) Method

```
[ILASM]
.method public hidebysig static class System.String[]
GetFiles(string path)

[C#]
public static string[] GetFiles(string path)
```

Summary

Returns the names of all files in the specified directory.

Parameters

Parameter	Description
<i>path</i>	A System.String containing the name of the directory for which file names are returned.

Return Value

A **System.String** array containing the names of the files in the specified directory.

Platforms that do not support this feature return **null**.

Description

This method is identical to **System.IO.Directory.GetFiles** (*path*, "*").

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory**.]

Exceptions

Exception	Condition
System.ArgumentNullException	<i>path</i> is null.
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid

	characters.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.IOException	<i>path</i> is a file name.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory and the files in that directory. See System.Security.Permissions.FileIOPermissionAccess and PathDiscovery .

5
6
7

1 Directory.GetFiles(System.String, 2 System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static class System.String[]  
5 GetFiles(string path, string searchPattern)  
  
6 [C#]  
7 public static string[] GetFiles(string path, string  
8 searchPattern)
```

9 Summary

10 Returns the names of files in the specified directory that match the
11 specified search pattern.

12 Parameters

13
14

Parameter	Description
<i>path</i>	A System.String containing the name of the directory to search.
<i>searchPattern</i>	A System.String containing the text pattern to match against the names of files in <i>path</i> . <i>searchPattern</i> cannot contain System.IO.Path.DirectorySeparatorChar or System.IO.Path.AltDirectorySeparatorChar .

15
16
17

16 Return Value

18 A **System.String** array containing the names of files in the specified
19 directory that match the specified search pattern.

20 Description

21 The following wild card specifiers are permitted in *searchPattern*:

Wild card	Description
*	Zero or more characters.
?	Exactly one character.

22
23
24
25
26
27
28

The period (".") character, if immediately followed by a wild card specifier, indicates that the period or the empty string matches the pattern. For example, "foo.*" and "foo.?" match "foo". Note that "foo.*" and "foo*" behave identically. If the period is not immediately followed by a wildcard, it has no special meaning (it represents a period).

1
2
3
4
5
6
7
8
9
10
11

Characters other than the wild card specifiers and the period always represent themselves, for example, the *searchPattern* string "*t" searches for all names in *path* ending with the letter "t". The *searchPattern* string "s*" searches for all names in *path* beginning with the letter "s".

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory.**]

12 **Exceptions**
13
14

Exception	Condition
System.ArgumentNullException	<i>searchPattern</i> or <i>path</i> is null .
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.IO.IOException	<i>path</i> is a file name.
System.Security.SecurityException	The caller does not have the required permission.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

15
16 **Permissions**
17
18

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory and the files in that directory. See System.Security.Permissions.FileIOPermissionAccess.PathDiscovery

19
20
21

1 Directory.GetFileSystemEntries(System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static class System.String[]  
5 GetFileSystemEntries(string path)  
  
6 [C#]  
7 public static string[] GetFileSystemEntries(string path)
```

8 Summary

9 Returns the names of all files and subdirectories in the specified
10 directory.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the name of the directory for which file and subdirectory names are returned.

14
15
16

15 Return Value

17 A **System.String** array containing the names of file system entries in
18 the specified directory.

19 Description

20 This method is identical to
21 **System.IO.Directory.GetFileSystemEntries** (*path*, "*").
22

23 The names returned by this method are prefixed with the directory
24 information provided in *path*. The *path* argument is permitted to
25 specify relative or absolute path information. Relative path information
26 is interpreted as relative to the current working directory. [Note: To
27 obtain the current working directory, see
28 **System.IO.Directory.GetCurrentDirectory**.]

29 Exceptions

30
31

Exception	Condition
System.ArgumentNullException	<i>path</i> is null .
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or

	more implementation-defined invalid characters.
System.Security.SecurityException	The caller does not have the required permission.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.IO.IOException	<i>path</i> is a file name.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory. See System.Security.Permissions.FileIOPermissionAccess: PathDiscovery

5
6
7

1 Directory.GetFileSystemEntries(System.String, System.String) Method

```
3 [ILASM]
4 .method public hidebysig static class System.String[]
5 GetFileSystemEntries(string path, string searchPattern)
6
7 [C#]
8 public static string[] GetFileSystemEntries(string path,
9 string searchPattern)
```

9 Summary

10 Returns an array of file and directory names matching the specified
11 search criteria.

12 Parameters

13
14

Parameter	Description
<i>path</i>	A System.String containing the name of the directory to search.
<i>searchPattern</i>	A System.String containing the text pattern for which to search. <i>searchPattern</i> cannot contain System.IO.Path.DirectorySeparatorChar or System.IO.Path.AltDirectorySeparatorChar .

15
16
17

16 Return Value

18 A **String** array containing file and directory names matching the
19 specified search criteria.

20 Description

21 The following wild card specifiers are permitted in *searchPattern*:

Wild card	Description
*	Zero or more characters.
?	Exactly one character.

22
23
24
25
26
27
28

The period (".") character, if immediately followed by a wild card specifier, indicates that the period or the empty string matches the pattern. For example, "foo.*" and "foo.?" match "foo". Note that "foo.*" and "foo*" behave identically. If the period is not immediately followed by a wildcard, it has no special meaning (it represents a period).

1
2
3
4
5
6
7
8
9
10
11
12

Characters other than the wild card specifiers represent themselves, for example, the *searchPattern* string "*t" searches for all names in *path* ending with the letter "t". The *searchPattern* string "s*" searches for all names in *path* beginning with the letter "s".

The names returned by this method are prefixed with the directory information provided in *path*. The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory.**]

13
14
15

Exceptions

Exception	Condition
System.ArgumentNullException	<i>searchPattern</i> or <i>path</i> is null .
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.Security.SecurityException	The caller does not have the required permission.
System.IO.DirectoryNotFoundException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.IO.IOException	<i>path</i> is a file name.

16
17
18
19

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to access path information for the specified directory. See System.Security.Permissions.FileIOPermissionAccess.PathDiscovery

20
21
22

1 Directory.GetLastAccessTime(System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static valuetype System.DateTime  
5 GetLastAccessTime(string path)  
  
6 [C#]  
7 public static DateTime GetLastAccessTime(string path)
```

8 Summary

9 Returns the date and time the specified file or directory was last
10 accessed.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the name of the file or directory for which to obtain access date and time information.

14
15
16

Return Value

17 A **System.DateTime** structure set to the date and time the specified
18 file or directory was last accessed. This value is expressed in local
19 time.

20
21 Platforms that do not support this feature return
22 **System.DateTime.MinValue**.

23 Description

24 This method is equivalent to **System.IO.File.GetLastAccessTime**
25 (*path*).

26
27 The *path* argument is permitted to specify relative or absolute path
28 information. Relative path information is interpreted as relative to the
29 current working directory. [Note: To obtain the current working
30 directory, see **System.IO.Directory.GetCurrentDirectory**.]

31 Exceptions

32
33

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string. contains only

	white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.IOException	The specified path was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to read the specified file or directory See System.Security.Permissions.FileIOPermissionAccess Read .

5
6
7

1 Directory.GetLastWriteTime(System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static valuetype System.DateTime  
5 GetLastWriteTime(string path)  
  
6 [C#]  
7 public static DateTime GetLastWriteTime(string path)
```

8 Summary

9 Returns the date and time the specified file or directory was last
10 written to.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the name of the file or directory for which to obtain modification date and time information.

14
15
16

Return Value

17 A **System.DateTime** structure set to the date and time the specified
18 file or directory was last written to. This value is expressed in local
19 time.

20
21
22

Platforms that do not support this feature return
System.DateTime.MinValue.

23 Description

24 This method is equivalent to **System.IO.File.GetLastWriteTime**
25 (*path*).

26
27
28
29
30

The *path* argument is permitted to specify relative or absolute path information. Relative path information is interpreted as relative to the current working directory. [Note: To obtain the current working directory, see **System.IO.Directory.GetCurrentDirectory**.]

31 Exceptions

32
33

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string. contains only

	white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.IOException	The specified path was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to read the specified file or directory See System.Security.Permissions.FileIOPermissionAccess Read .

5
6
7

1 Directory.Move(System.String, 2 System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static void Move(string  
5 sourceDirName, string destDirName)  
  
6 [C#]  
7 public static void Move(string sourceDirName, string  
8 destDirName)
```

9 Summary

10 Moves a file or a directory and its contents to a new location.

11 Parameters

12
13

Parameter	Description
<i>sourceDirName</i>	A System.String containing the name of the file or directory to move.
<i>destDirName</i>	A System.String containing the new location for <i>sourceDirName</i> . This string cannot identify an existing file or directory.

14
15

Description

16 The *destDirName* argument cannot specify a location on a different
17 disk or volume than *sourceDirName*. The *sourceDirName* and
18 *destDirName* arguments cannot identify the same file or directory.

19
20 [Note: This method throws a **System.IO.IOException** if, for
21 example, you try to move "\mydir" to "\public", and "\public" already
22 exists. You must specify "\public\mydir" as the *destDirName*.]
23

24 The *sourceDirName* and *destDirName* arguments are permitted to
25 specify relative or absolute path information. Relative path information
26 is interpreted as relative to the current working directory. [Note: To
27 obtain the current working directory, see
28 **System.IO.Directory.GetCurrentDirectory**.]

29 Exceptions

30
31

Exception	Condition
System.ArgumentException	<i>sourceDirName</i> or <i>destDirName</i> is a zero-length string, contains only white space. or contains implementation-

	defined invalid characters.
System.ArgumentNullException	<i>sourceDirName</i> or <i>destDirName</i> is null .
System.Security.SecurityException	The caller does not have the required permission.
System.IO.IOException	An attempt was made to move a directory to a different volume, or <i>destDirName</i> already exists.
System.IO.DirectoryNotFoundException	<i>sourceDirName</i> was not found.
System.IO.PathTooLongException	The length or absolute path information for <i>sourceDirName</i> or <i>destDirName</i> exceeds the system-defined maximum length.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to read from <i>sourceDirName</i> , and write to <i>sourceDirName</i> and <i>destDirName</i> . See System.Security.Permissions.FileIOPermissionAccess.Read , System.Security.Permissions.FileIOPermissionAccess.Write .

5
6
7

1 Directory.SetCreationTime(System.String, 2 System.DateTime) Method

```
3 [ILASM]  
4 .method public hidebysig static void SetCreationTime(string  
5 path, valuetype System.DateTime creationTime)  
  
6 [C#]  
7 public static void SetCreationTime(string path, DateTime  
8 creationTime)
```

9 Summary

10 Sets the creation date and time for the specified file or directory.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the name of the file or directory for which to set the creation date and time information.
<i>creationTime</i>	A System.DateTime containing the value to set for the creation date and time of <i>path</i> . This value is expressed in local time.

14
15

Description

16 The *path* argument is permitted to specify relative or absolute path
17 information. Relative path information is interpreted as relative to the
18 current working directory. [Note: To obtain the current working
19 directory, see **System.IO.Directory.GetCurrentDirectory**.]

20
21
22
23

On platforms that do not support this feature, this method has no effect. If this feature is supported, the range of dates that is valid for this operation is implementation-specific.

24 Exceptions

25
26

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentOutOfRangeException	<i>creationTime</i> specifies a value outside the range of date/times permitted for this operation.

System.ArgumentNullException	<i>path</i> is null .
System.IO.FileNotFoundException	<i>path</i> was not found.
System.IO.IOException	An I/O error occurred while performing the operation.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to write to the specified file or directory. See System.Security.Permissions.FileIOPermissionAccess.Write .

5
6
7

1 Directory.SetCurrentDirectory(System.String) Method

```
3 [ILASM]  
4 .method public hidebysig static void  
5 SetCurrentDirectory(string path)  
  
6 [C#]  
7 public static void SetCurrentDirectory(string path)
```

8 Summary

9 Sets the application's current working directory to the specified
10 directory.

11 Parameters

12
13

Parameter	Description
<i>path</i>	A System.String containing the path to which the current working directory is set.

14
15

15 Description

16 When the application terminates, the working directory is restored to
17 its original location (the directory where the process was started).

18
19 The *path* argument is permitted to specify relative or absolute path
20 information. Relative path information is interpreted as relative to the
21 current working directory. [Note: To obtain the current working
22 directory, see **System.IO.Directory.GetCurrentDirectory.**]

23
24 On platforms that do not support this feature, this method has no
25 effect.

26 Exceptions

27
28

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.FileNotFoundException	<i>path</i> was not found.
System.IO.IOException	An I/O error occurred while performing the operation.

1
2
3

System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the system-defined maximum length.
---------------------------------------	---

1 Directory.SetLastAccessTime(System.String, System.DateTime) Method

```
3 [ILASM]
4 .method public hidebysig static void
5 SetLastAccessTime(string path, valuetype System.DateTime
6 lastAccessTime)
7
8 [C#]
9 public static void SetLastAccessTime(string path, DateTime
lastAccessTime)
```

10 Summary

11 Sets the date and time the specified file or directory was last accessed.

12 Parameters

13
14

Parameter	Description
<i>path</i>	A System.String containing the name of the file or directory for which to set the access date and time information.
<i>lastAccessTime</i>	A System.DateTime containing the value to set for the access date and time of <i>path</i> . This value is expressed in local time.

15
16

16 Description

17 The *path* argument is permitted to specify relative or absolute path
18 information. Relative path information is interpreted as relative to the
19 current working directory. [*Note:* To obtain the current working
20 directory, see **System.IO.Directory.GetCurrentDirectory.**]

21
22
23
24

On platforms that do not support this feature, this method has no effect. If this feature is supported, the range of dates that is valid for this operation is implementation-specific.

25 Exceptions

26
27

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.IOException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path

1
2
3
4

Permissions

	information for <i>path</i> exceeds the system-defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

5
6
7

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to write to the specified file or directory. See System.Security.Permissions.FileIOPermissionAccess . Write .

1 Directory.SetLastWriteTime(System.String, System.DateTime) Method

```
3 [ILASM]
4 .method public hidebysig static void
5 SetLastWriteTime(string path, valuetype System.DateTime
6 lastWriteTime)
7
8 [C#]
9 public static void SetLastWriteTime(string path, DateTime
lastWriteTime)
```

10 Summary

11 Sets the date and time a directory was last written to.

12 Parameters

13
14

Parameter	Description
<i>path</i>	A System.String containing the name of the directory for which to set the date and time information.
<i>lastWriteTime</i>	A System.DateTime containing the value to set for the last write date and time of <i>path</i> . This value is expressed in local time.

15
16

16 Description

17 Relative path information is interpreted as relative to the current
18 working directory. [Note: To obtain the current working directory, see
19 **System.IO.Directory.GetCurrentDirectory**.]
20

21 On platforms that do not support this feature, this method has no
22 effect. If this feature is supported, the range of dates that is valid for
23 this operation is implementation-specific.

24 Exceptions

25
26

Exception	Condition
System.ArgumentException	<i>path</i> is a zero-length string, contains only white space, or contains one or more implementation-defined invalid characters.
System.ArgumentNullException	<i>path</i> is null .
System.IO.IOException	<i>path</i> was not found.
System.IO.PathTooLongException	The length of <i>path</i> or the absolute path information for <i>path</i> exceeds the svstem-

	defined maximum length.
System.Security.SecurityException	The caller does not have the required permission.

1
2
3
4

Permissions

Permission	Description
System.Security.Permissions.FileIOPermission	Requires permission to write to the specified file. See System.Security.Permissions.FileIOPermissionAccess.Write .

5
6