JAVA.IO.FILE CLASS

Introduction

The **Java.io.File** class is an abstract representation of file and directory pathnames. Following are the important points about File:

- Instances may or may not denote an actual file-system object such as a file or a directory. If it does denote such an object then that object resides in a partition. A partition is an operating system-specific portion of storage for a file system.
- A file system may implement restrictions to certain operations on the actual file-system object, such as reading, writing, and executing. These restrictions are collectively known as access permissions.
- Instances of the File class are immutable; that is, once created, the abstract pathname represented by a File object will never change.

Class declaration

Following is the declaration for **Java.io.File** class:

```
public class File
  extends Object
    implements Serializable, Comparable<File>
```

Field

Following are the fields for **Java.io.File** class:

- **static String pathSeparator** -- This is the system-dependent path-separator character, represented as a string for convenience.
- **static char pathSeparatorChar** -- This is the system-dependent path-separator character.
- **static String separator** -- This is the system-dependent default name-separator character, represented as a string for convenience.
- **static char separatorChar** -- This is the system-dependent default name-separator character.

Class constructors

S.N.	Constructor & Description
1	**File***Fileparent, Stringchild* This method creates a new File instance from a parent abstract pathname and a child pathname string.
2	**File***Stringpathname* This method creates a new File instance by converting the given pathname string into an abstract pathname.
3	**File***Stringparent, Stringchild*

This method creates a new File instance from a parent pathname string and a child pathname string.

4

File*URIuri*

This method Creates a new File instance by converting the given file : URI into an abstract pathname.

Class methods

S.N.	Method & Description
1	boolean canExecute This method tests whether the application can execute the file denoted by this abstract pathname.
2	boolean canRead This method tests whether the application can read the file denoted by this abstract pathname.
3	boolean canWrite This method tests whether the application can modify the file denoted by this abstract pathname.
4	int compareTo*Filepathname* This method compares two abstract pathnames lexicographically.
5	boolean createNewFile This method atomically creates a new, empty file named by this abstract pathname if and only if a file with this name does not yet exist.
6	static File createTempFile*Stringprefix, Stringsuffix* This method creates an empty file in the default temporary-file directory, using the given prefix and suffix to generate its name.
7	static File createTempFile*Stringprefix, Stringsuffix, Filedirectory* This method Creates a new empty file in the specified directory, using the given prefix and suffix strings to generate its name.
8	boolean delete This method deletes the file or directory denoted by this abstract pathname.

9

void deleteOnExit

This method requests that the file or directory denoted by this abstract pathname be deleted when the virtual machine terminates.

10

boolean equals*Objectobj*

This method tests this abstract pathname for equality with the given object.

11

boolean exists

This method tests whether the file or directory denoted by this abstract pathname exists.

12

File getAbsoluteFile

This method returns the absolute form of this abstract pathname.

13

String getAbsolutePath

This method returns the absolute pathname string of this abstract pathname.

14

File getCanonicalFile

This method returns the canonical form of this abstract pathname.

15

String getCanonicalPath

This method returns the canonical pathname string of this abstract pathname.

16

long getFreeSpace

This method returns the number of unallocated bytes in the partition named by this abstract path name.

17

String getName

This method returns the name of the file or directory denoted by this abstract pathname.

18

String getParent

This method returns the pathname string of this abstract pathname's parent, or null if this pathname does not name a parent directory.

19

File getParentFile

This method returns the abstract pathname of this abstract pathname's parent, or null if this pathname does not name a parent directory.

20

String getPath

This method converts this abstract pathname into a pathname string.

21

long getTotalSpace

This method returns the size of the partition named by this abstract pathname.

22

long getUsableSpace

This method returns the number of bytes available to this virtual machine on the partition named by this abstract pathname.

23

int hashCode

This method computes a hash code for this abstract pathname.

24

boolean isAbsolute

This method tests whether this abstract pathname is absolute.

25

boolean isDirectory

This method tests whether the file denoted by this abstract pathname is a directory.

26

boolean isFile

This method tests whether the file denoted by this abstract pathname is a normal file.

27

boolean isHidden

This method tests whether the file named by this abstract pathname is a hidden file.

28

long lastModified

This method returns the time that the file denoted by this abstract pathname was last modified.

29

long length

This method returns the length of the file denoted by this abstract pathname.

30

String[] list

This method returns an array of strings naming the files and directories in the directory denoted by this abstract pathname.

31

String[] list*FilenameFilterfilter*

This method returns an array of strings naming the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter.

32

File[] listFiles

This method returns an array of abstract pathnames denoting the files in the directory denoted by this abstract pathname.

33

File[] listFiles*FileFilterfilter*

This method returns an array of abstract pathnames denoting the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter.

34

File[] listFiles*FilenameFilterfilter*

This method returns an array of abstract pathnames denoting the files and directories in the directory denoted by this abstract pathname that satisfy the specified filter.

35

static File[] listRoots

This method lists the available filesystem roots.

36

boolean mkdir

This method creates the directory named by this abstract pathname.

37

boolean mkdirs

This method creates the directory named by this abstract pathname, including any necessary but non existent parent directories.

38

boolean renameTo*Filedest*

This method renames the file denoted by this abstract pathname.

39

boolean setExecutable*booleanexecutable*

This is a convenience method to set the owner's execute permission for this abstract pathname.

40

boolean setExecutable*booleanexecutable, booleanownerOnly*

This method Sets the owner's or everybody's execute permission for this abstract pathname.

41

boolean setLastModified*longtime*

This method sets the last-modified time of the file or directory named by this abstract

pathname.

42

boolean setReadable*booleanreadable*

This is a convenience method to set the owner's read permission for this abstract pathname.

43

boolean setReadable*booleanreadable, booleanownerOnly*

This method sets the owner's or everybody's read permission for this abstract pathname.

44

boolean setReadOnly

This method marks the file or directory named by this abstract pathname so that only read operations are allowed.

45

boolean setWritable*booleanwritable*

This is a convenience method to set the owner's write permission for this abstract pathname.

46

boolean setWritable*booleanwritable, booleanownerOnly*

This method sets the owner's or everybody's write permission for this abstract pathname.

47

String toString

This method returns the pathname string of this abstract pathname.

48

URI toURI

This method constructs a file : URI that represents this abstract pathname.

Methods inherited

This class inherits methods from the following classes:

- Java.io.Object