**File class in Java**

In Java, a File is an abstract data type. A named location used to store related information is known as a File. There are several File Operations like creating a new File, getting information about File, writing into a File, reading from a File and deleting a File.

**File Methods**

* public String getName()
* public String getPath()
* public String getAbsolutePath()
* public String getParent()
* public boolean exists() throws SecurityException
* public boolean canWrite() throws SecurityException
* public boolean canRead() throws SecurityException
* public boolean isFile() throws SecurityException
* public boolean isDirectory() throws SecurityException
* public boolean isAbsolute()
* public long lastModified() throws SecurityException
* public long length() throws SecurityException
* public boolean mkdir()
* public boolean mkdirs() throws SecurityException
* public boolean renameTo(File destination) throws SecurityException
* public boolean delete() throws SecurityException

import java.io.\*;

public class FileInfo {

public static void main(String[] args) throws IOException{

try{

File f = new File("e:/abc.txt");

f.createNewFile();

if (f.exists()) {

System.out.println("getName: " + f.getName());

System.out.println("getPath: " + f.getPath());

System.out.println("getAbsolutePath: " + f.getAbsolutePath());

System.out.println("getParent: " + f.getParent());

if (f.isFile()) {

System.out.println(f.getName() + " is a file.");

}

if (f.isDirectory()) {

System.out.println(f.getName() + " is a directory.");

}

else {

System.out.println("What is this?");

}

if (f.isAbsolute()) {

System.out.println(f.getName() + " is an absolute path.");

}

else {

System.out.println(f.getName() + " is not an absolute path."); }

try {

System.out.println("Last Modified" + f.lastModified());

System.out.println(f.getName() + " is " + f.length() + " bytes.");

}

catch (Exception ex) {System.out.println(ex);}

}catch(Exception ex){ex.printStackTrace();}

}

}

}

**Example of How to Create Directory in Java**

Just like above example of creating file in Java we can create directory in Java, only difference is that we need to use mkdir() method to create directory in Java

import java.io.\*;

public class DirectoryExample {

public static void main(String[] args) {

boolean dirFlag = false;

// create File object

File stockDir = new File("d://Stock/ stockDir ");

try { dirFlag = stockDir.mkdir();

} catch (SecurityException Se) {

System.out.println(Se);}

if (dirFlag)

System.out.println("Directory created successfully");

else

System.out.println("Directory was not created successfully");

}}

**Instantiating a java.io.File**

File file = new File("c:\\data\\input-file.txt");

**Check if File Exists**

Once you have instantiated a File object you can check if the corresponding file actually exists already. The File class constructor will not fail if the file does not already exists. To check if the file exists, call the exists() method.

File file = new File("c:\\data\\input-file.txt");

boolean fileExists = file.exists();

**Create a Directory if it Does Not Exist**

The mkdir() method creates a single directory if it does not already exist.

The mkdirs() method creates a multiple directory if it does not already exist.

public static void main(String args[])

{

boolean dirFlag = false;

// create File object

File packDir = new File("e:/mypack/parkDir");

try { dirFlag = packDir.mkdir();}

catch (SecurityException Se) {System.out.println(Se);}

if (dirFlag)

System.out.println("Directory created successfully");

else

System.out.println("Directory was not created successfully");

}

**File Length**

To read the length of a file in bytes, call the length() method.

File file = new File("c:\\data\\input-file.txt");

long length = file.length();

**Rename or Move File**

To rename (or move) a file, call the method renameTo() on the File class.

File file = new File("c:\\data\\input-file.txt");

boolean success = file.renameTo(new File("c:\\data\\new-file.txt"));

**Delete File**

To delete a file call the delete() method. Here is a simple example:

File file = new File("c:\\data\\input-file.txt");

boolean success = file.delete();

**Check if Path is File or Directory**

File file = new File("c:\\data");

boolean isDirectory = file.isDirectory();

**Read List of Files in Directory**

File file = new File("c:\\data");

String[] fileNames = file.list();

File[] files = file.listFiles();