**Assisted Practice: 3.7 File Handling Mechanisms**

This section will guide you to:

* Write a program in Java to demonstrate exception handling
* Use Eclipse (the popular text editor for Java programs)
* Push code to Git

This lab has six sub-sections, namely:

* + 1. Creating a new project in Eclipse
    2. Writing a program in Java to create a file
    3. Writing a program in Java to read a file
    4. Writing a program in Java to update a file
    5. Writing a program in Java to delete a file
    6. Pushing the code to your GitHub repositories

**Step 3.7.1:** Creating a new project in Eclipse

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next.
* Type in any project name and click on “Finish.”
* Select your project and go to File -> New -> Class.
* Enter **CreateNewFile** in any class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

**Step 3.7.2:** Writing a program in Java to create a file

import java.io.File;

import java.io.FileOutputStream;

import java.io.FileWriter;

import java.io.IOException;

import java.nio.charset.StandardCharsets;

import java.nio.file.Files;

import java.nio.file.Paths;

import java.nio.file.StandardOpenOption;

import java.util.Arrays;

import java.util.List;

public class CreateNewFile

{

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

{

createFileUsingFileClass();

createFileUsingFileOutputStreamClass();

createFileIn\_NIO();

}

private static void createFileUsingFileClass() throws IOException

{

File file = new File("c://temp//testFile1.txt");

//Create the file

if (file.createNewFile()){

System.out.println("File is created!");

}else{

System.out.println("File already exists.");

}

//Write Content

FileWriter writer = new FileWriter(file);

writer.write("Test data");

writer.close();

}

private static void createFileUsingFileOutputStreamClass() throws IOException

{

String data = "Test data";

FileOutputStream out = new FileOutputStream("c://temp//testFile2.txt");

out.write(data.getBytes());

out.close();

}

private static void createFileIn\_NIO() throws IOException

{

String data = "Test data";

Files.write(Paths.get("c://temp//testFile3.txt"), data.getBytes());

List<String> lines = Arrays.asList("1st line", "2nd line")

Files.write(Paths.get("file6.txt"),

lines,

StandardCharsets.UTF\_8,

StandardOpenOption.CREATE,

StandardOpenOption.APPEND);

}

}

**Output:**



**Step 3.7.3:** Writing a program in Java to read a file

* Select your project and go to File -> New -> Class.
* Enter **ReadFileIntoList** in any class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

import java.util.\*;

import java.nio.charset.StandardCharsets;

import java.nio.file.\*;

import java.io.\*;

public class ReadFileIntoList

{

public static List<String> readFileInList(String fileName)

{

List<String> lines = Collections.emptyList();

try

{

lines =

Files.readAllLines(Paths.get(fileName), StandardCharsets.UTF\_8);

}

catch (IOException e)

{

e.printStackTrace();

}

return lines;

}

public static void main(String[] args)

{

List l = readFileInList("c://temp//testFile2.txt");

Iterator<String> itr = l.iterator();

while (itr.hasNext())

System.out.println(itr.next());

}

}

**Step 3.7.4:** Writing a program in Java to update a file

* Select your project and go to File -> New -> Class.
* Enter **TextFileModificationProgram** in any class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

public class TextFileModificationProgram

{

static void modifyFile(String filePath, String oldString, String newString)

{

File fileToBeModified = new File(filePath);

String oldContent = "";

BufferedReader reader = null;

FileWriter writer = null;

try

{

reader = new BufferedReader(new FileReader(fileToBeModified));

String line = reader.readLine();

while (line != null)

{

oldContent = oldContent + line + System.lineSeparator();

line = reader.readLine();

}

String newContent = oldContent.replaceAll(oldString, newString);

writer = new FileWriter(fileToBeModified);

writer.write(newContent);

}

catch (IOException e)

{

e.printStackTrace();

}

finally

{

try

{

reader.close();

writer.close();

}

catch (IOException e)

{

e.printStackTrace();

}

}

}

public static void main(String[] args)

{

modifyFile("c://temp//testFile2.txt", "85", "95");

System.out.println("done");

}

}

**Output:**



**Step 3.7.5:** Writing a program in Java to delete a file

* Select your project and go to File -> New -> Class.
* Enter **Test** in any class name, check the checkbox “public static void main(String[] args)”., and click on “Finish.”

import java.io.IOException;

import java.nio.file.\*;

public class Test

{

public static void main(String[] args)

{

try

{

Files.deleteIfExists(Paths.get("c://temp//testFile2.txt"));

}

catch(NoSuchFileException e)

{

System.out.println("No such file/directory exists");

}

catch(DirectoryNotEmptyException e)

{

System.out.println("Directory is not empty.");

}

catch(IOException e)

{

System.out.println("Invalid permissions.");

}

System.out.println("Deletion successful.");

}

}

**Output:**



**Step 3.7.6:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**