Object Oriented Programming in Java

(Report)

Submitted by - Zeeshan Sharif
E.No. - 2020BITE’012

File Handling Concepts used in Java Program.

Introduction:

This report aims to explain the file handling concepts used in the Java program we did. The program performs various tasks related to file handling, including directory creation, file creation, writing and appending new texts, reading file contents, using both **byte and character streams, and handling exceptions.**

1) Directory Creation

The program demonstrates the creation of a directory using the `mkdir()` method of the `File` class. The `mkdir()` method creates a new directory if it doesn't already exist. If the directory creation is successful, it displays a message indicating the directory path. Otherwise, it handles the case where the directory creation fails.

2) File Creation

The program creates a text file inside the previously created directory using the `createNewFile()` method of the `File` class.

This method creates a new file if it doesn't exist. If the file creation is successful, it displays a message indicating the file path. If the file already exists, it displays a corresponding message.

3) Writing File Contents

The program writes sample text content to the file using character streams. It utilizes the `FileWriter` class to write data to the file. The `write()` method of the `FileWriter` class is used to write the text content to the file. In case of any exceptions during the write operation, proper error handling is implemented.

4) Reading File Contents

The program reads the contents of the file using character streams. It employs the `FileReader` class to read the data from the file. The `read()` method of the `FileReader` class is used to read the file content character by character. The read characters are then appended to a `StringBuilder` to form the complete content. The program displays the file content on the console. Appropriate error handling is implemented to handle exceptions that may occur during the read operation.

5) Writing and Reading Using Byte Streams

The program extends its functionality by demonstrating the usage of byte streams. It creates a new text file and writes text content using byte streams. The `FileOutputStream` class is used to write data in bytes to the file. The program also reads the contents of the byte file using the

`FileInputStream` class and displays the content on the console. Proper error handling is implemented for these operations as well.

6) Exception Handling

Throughout the program, **error handling and exception handling mechanisms are implemented using try-catch blocks.** It ensures that any potential errors or exceptions, such as file or directory not found, I/O exceptions, etc., are properly caught and handled. Detailed error messages are displayed to provide information about the encountered issues.

Conclusion

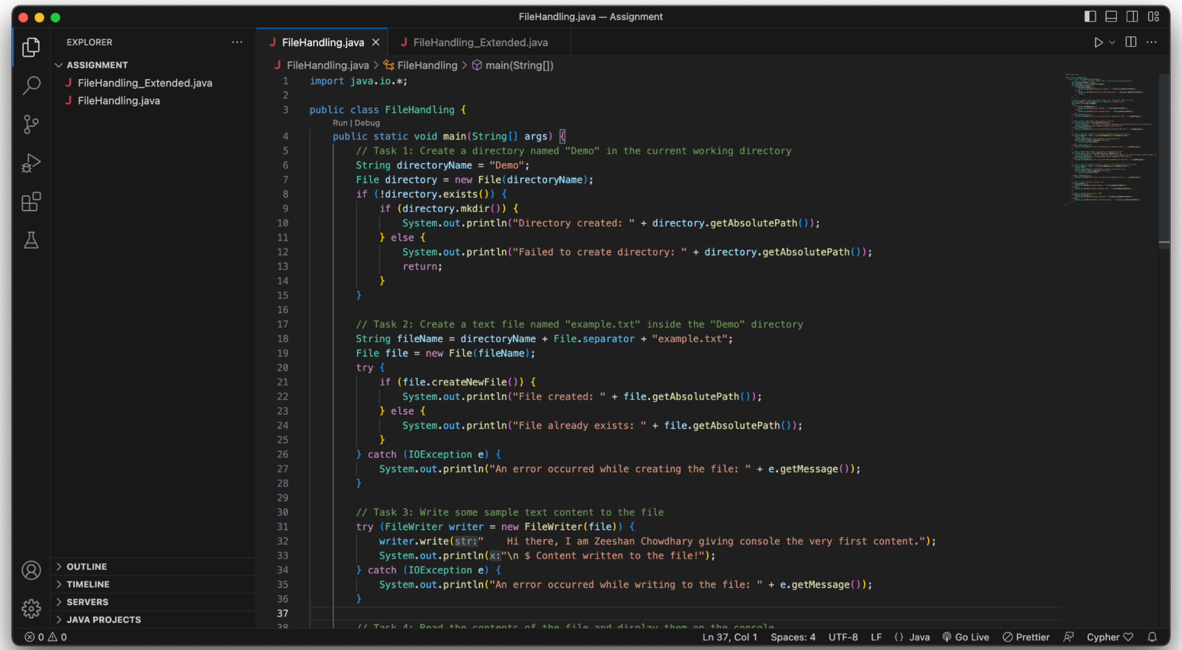
The Java program effectively demonstrates various file handling concepts such as directory creation, file creation, writing and reading file contents using both byte and character streams. It also showcases the implementation of error handling and exception handling mechanisms to handle potential issues during file handling operations.

Q1. File Handling and Stream Handling

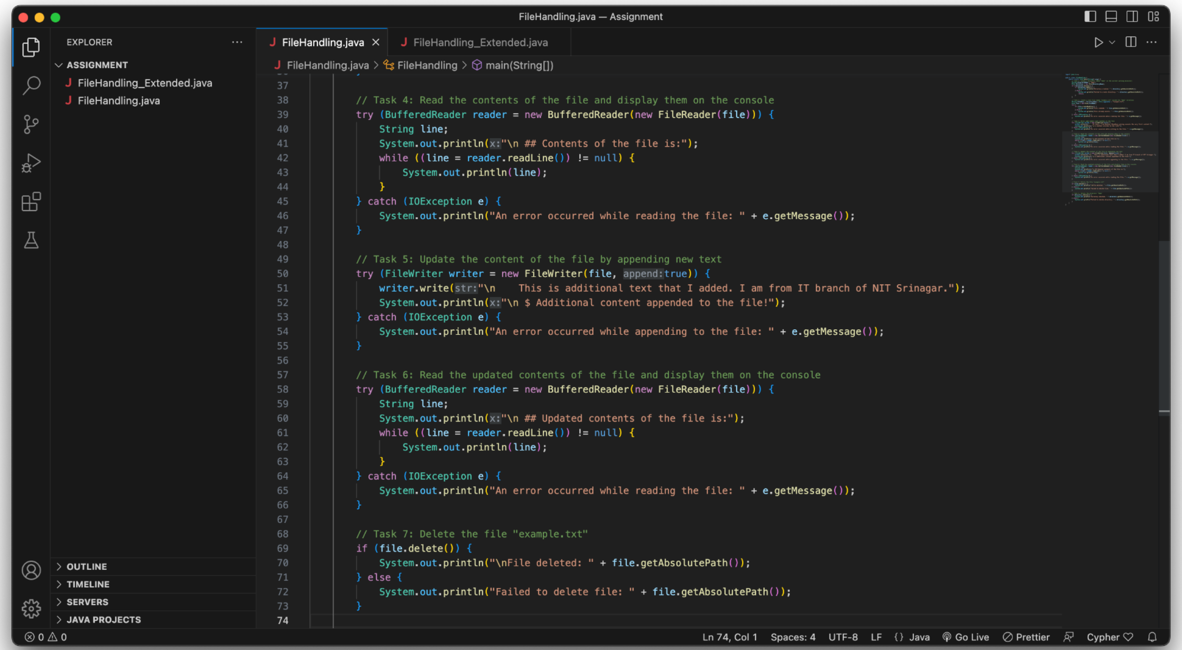
Create a Java program that performs the following tasks:

1. Create a directory named "Demo" in the current working directory.
2. Create a text file named "example.txt" inside the "Demo" directory.
3. Write some sample text content to the file.
4. Read the contents of the file and display them on the console.
5. Update the content of the file by appending new text.
6. Read the updated contents of the file and display them on the console.
7. Delete the file "example.txt"
8. Delete the directory "Demo"

CODE </>

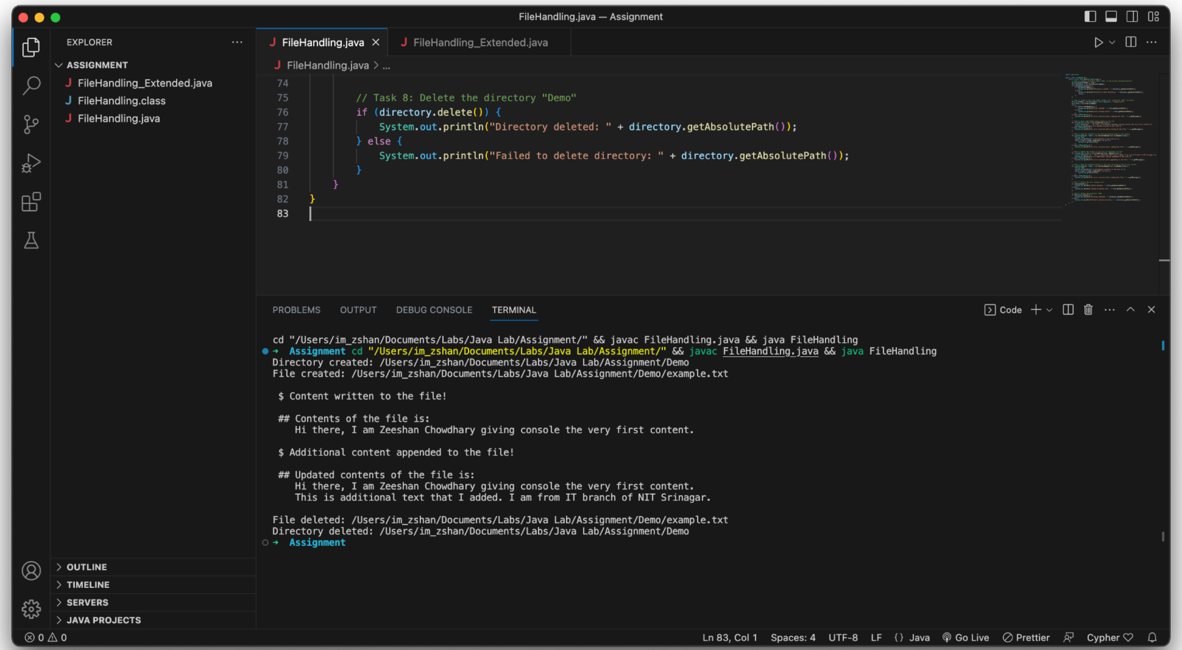

FileHandling.java — Assignment
EXPLORER
ASSIGNMENT
FileHandling_Extended.java
FileHandling.java
FileHandling.java
FileHandling_Extended.java
FileHandling.java > FileHandling > main(String[])
OUTLINE
TIMELINE
SERVERS
JAVA PROJECTS
Ln 37, Col 1 Spaces: 4 UTF-8 LF {} Java Go Live Prettier Cypher

```java
import java.io.*;

public class FileHandling {
    Run | Debug
    public static void main(String[] args) {
        // Task 1: Create a directory named "Demo" in the current working directory
        String directoryName = "Demo";
        File directory = new File(directoryName);
        if (!directory.exists()) {
            if (directory.mkdir()) {
                System.out.println("Directory created: " + directory.getAbsolutePath());
            } else {
                System.out.println("Failed to create directory: " + directory.getAbsolutePath());
                return;
            }
        }

        // Task 2: Create a text file named "example.txt" inside the "Demo" directory
        String fileName = directoryName + File.separator + "example.txt";
        File file = new File(fileName);
        try {
            if (file.createNewFile()) {
                System.out.println("File created: " + file.getAbsolutePath());
            } else {
                System.out.println("File already exists: " + file.getAbsolutePath());
            }
        } catch (IOException e) {
            System.out.println("An error occurred while creating the file: " + e.getMessage());
        }

        // Task 3: Write some sample text content to the file
        try (FileWriter writer = new FileWriter(file)) {
            writer.write(str:"    Hi there, I am Zeeshan Chowdhary giving console the very first content.");
            System.out.println(x:"\n $ Content written to the file!");
        } catch (IOException e) {
            System.out.println("An error occurred while writing to the file: " + e.getMessage());
        }

        // Task 4: Read the contents of the file and display them on the console
```



```java
        // Task 4: Read the contents of the file and display them on the console
        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            System.out.println(x:"\n ## Contents of the file is:");
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file: " + e.getMessage());
        }

        // Task 5: Update the content of the file by appending new text
        try (FileWriter writer = new FileWriter(file, append:true)) {
            writer.write(str:"\n    This is additional text that I added. I am from IT branch of NIT Srinagar.");
            System.out.println(x:"\n $ Additional content appended to the file!");
        } catch (IOException e) {
            System.out.println("An error occurred while appending to the file: " + e.getMessage());
        }

        // Task 6: Read the updated contents of the file and display them on the console
        try (BufferedReader reader = new BufferedReader(new FileReader(file))) {
            String line;
            System.out.println(x:"\n ## Updated contents of the file is:");
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file: " + e.getMessage());
        }

        // Task 7: Delete the file "example.txt"
        if (file.delete()) {
            System.out.println("\nFile deleted: " + file.getAbsolutePath());
        } else {
            System.out.println("Failed to delete file: " + file.getAbsolutePath());
        }

```

Output:



```java

        // Task 8: Delete the directory "Demo"
        if (directory.delete()) {
            System.out.println("Directory deleted: " + directory.getAbsolutePath());
        } else {
            System.out.println("Failed to delete directory: " + directory.getAbsolutePath());
        }
    }
}
```

```
cd "/Users/im_zshan/Documents/Labs/Java Lab/Assignment/" && javac FileHandling.java && java FileHandling
● → Assignment cd "/Users/im_zshan/Documents/Labs/Java Lab/Assignment/" && javac FileHandling.java && java FileHandling
Directory created: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo
File created: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo/example.txt

 $ Content written to the file!

 ## Contents of the file is:
    Hi there, I am Zeeshan Chowdhary giving console the very first content.

 $ Additional content appended to the file!

 ## Updated contents of the file is:
    Hi there, I am Zeeshan Chowdhary giving console the very first content.
    This is additional text that I added. I am from IT branch of NIT Srinagar.

File deleted: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo/example.txt
Directory deleted: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo
○ → Assignment
```

Now, we’ ll extend the program to perform the following additional tasks:

1. Create a new text file named "byteExample.txt" and write some text using byte streams.

2. Read the contents of "byteExample.txt" using byte streams and display them on the console.

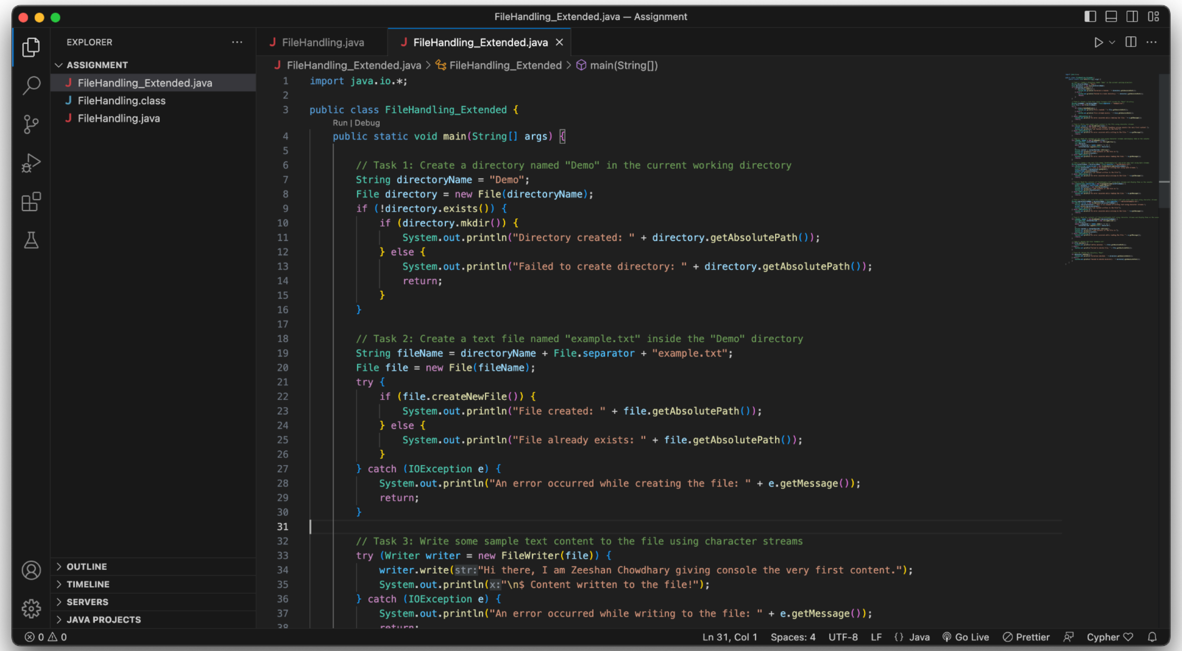
3. Create a new text file named "characterExample.txt" and write some text using character streams.

4. Read the contents of "characterExample.txt" using character streams and display them on the console.

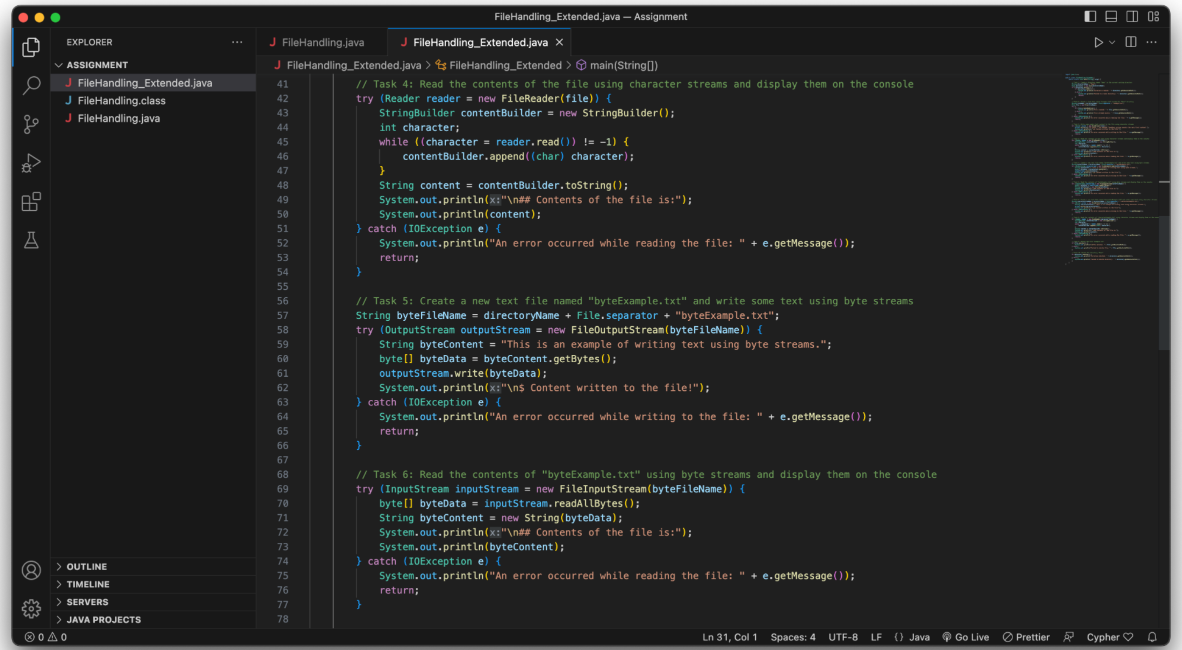
5. Ensure error handling and exception handling mechanisms are implemented throughout the program.

Note - Handle situations such as file or directory not found, I/O exceptions, etc.

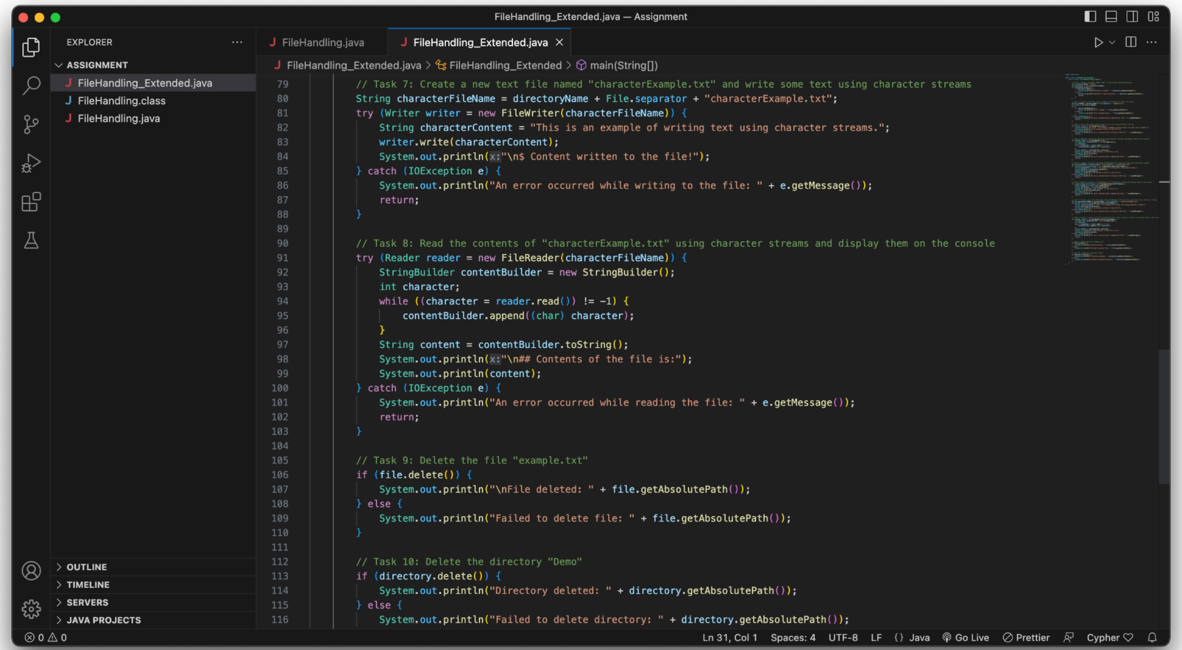
CODE </>



```java
import java.io.*;

public class FileHandling_Extended {
    public static void main(String[] args) {

        // Task 1: Create a directory named "Demo" in the current working directory
        String directoryName = "Demo";
        File directory = new File(directoryName);
        if (!directory.exists()) {
            if (directory.mkdir()) {
                System.out.println("Directory created: " + directory.getAbsolutePath());
            } else {
                System.out.println("Failed to create directory: " + directory.getAbsolutePath());
                return;
            }
        }

        // Task 2: Create a text file named "example.txt" inside the "Demo" directory
        String fileName = directoryName + File.separator + "example.txt";
        File file = new File(fileName);
        try {
            if (file.createNewFile()) {
                System.out.println("File created: " + file.getAbsolutePath());
            } else {
                System.out.println("File already exists: " + file.getAbsolutePath());
            }
        } catch (IOException e) {
            System.out.println("An error occurred while creating the file: " + e.getMessage());
            return;
        }

        // Task 3: Write some sample text content to the file using character streams
        try (Writer writer = new FileWriter(file)) {
            writer.write(str:"Hi there, I am Zeeshan Chowdhary giving console the very first content.");
            System.out.println(x:"\n$ Content written to the file!");
        } catch (IOException e) {
            System.out.println("An error occurred while writing to the file: " + e.getMessage());
```

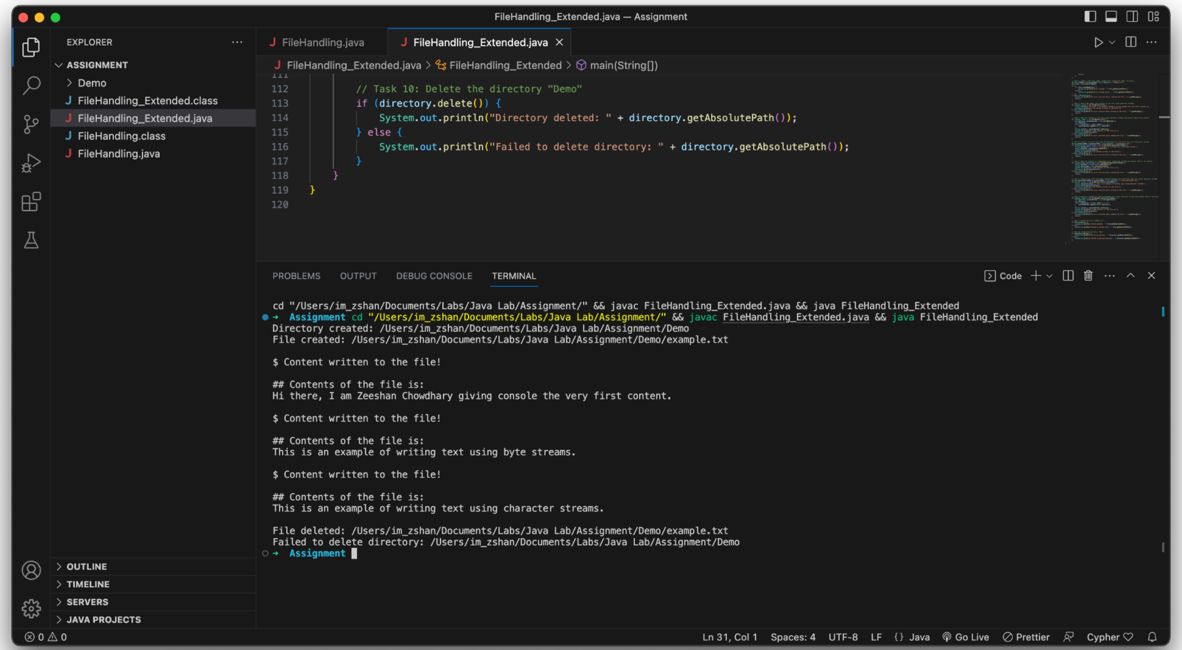

FileHandling_Extended.java — Assignment
EXPLORER
ASSIGNMENT
FileHandling_Extended.java
FileHandling.class
FileHandling.java
FileHandling.java
FileHandling_Extended.java
FileHandling_Extended.java > FileHandling_Extended > main(String[])
OUTLINE
TIMELINE
SERVERS
JAVA PROJECTS
0 0
Ln 31, Col 1 Spaces: 4 UTF-8 LF Java Go Live Prettier Cypher

```java
        // Task 4: Read the contents of the file using character streams and display them on the console
        try (Reader reader = new FileReader(file)) {
            StringBuilder contentBuilder = new StringBuilder();
            int character;
            while ((character = reader.read()) != -1) {
                contentBuilder.append((char) character);
            }
            String content = contentBuilder.toString();
            System.out.println(x:"\n## Contents of the file is:");
            System.out.println(content);
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file: " + e.getMessage());
            return;
        }

        // Task 5: Create a new text file named "byteExample.txt" and write some text using byte streams
        String byteFileName = directoryName + File.separator + "byteExample.txt";
        try (OutputStream outputStream = new FileOutputStream(byteFileName)) {
            String byteContent = "This is an example of writing text using byte streams.";
            byte[] byteData = byteContent.getBytes();
            outputStream.write(byteData);
            System.out.println(x:"\n$ Content written to the file!");
        } catch (IOException e) {
            System.out.println("An error occurred while writing to the file: " + e.getMessage());
            return;
        }

        // Task 6: Read the contents of "byteExample.txt" using byte streams and display them on the console
        try (InputStream inputStream = new FileInputStream(byteFileName)) {
            byte[] byteData = inputStream.readAllBytes();
            String byteContent = new String(byteData);
            System.out.println(x:"\n## Contents of the file is:");
            System.out.println(byteContent);
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file: " + e.getMessage());
            return;
        }

```


FileHandling_Extended.java — Assignment
EXPLORER
ASSIGNMENT
FileHandling_Extended.java
FileHandling.class
FileHandling.java
FileHandling.java
FileHandling_Extended.java
FileHandling_Extended.java > FileHandling_Extended > main(String[])
OUTLINE
TIMELINE
SERVERS
JAVA PROJECTS
0 0
Ln 31, Col 1 Spaces: 4 UTF-8 LF Java Go Live Prettier Cypher

```java
        // Task 7: Create a new text file named "characterExample.txt" and write some text using character streams
        String characterFileName = directoryName + File.separator + "characterExample.txt";
        try (Writer writer = new FileWriter(characterFileName)) {
            String characterContent = "This is an example of writing text using character streams.";
            writer.write(characterContent);
            System.out.println(x:"\n$ Content written to the file!");
        } catch (IOException e) {
            System.out.println("An error occurred while writing to the file: " + e.getMessage());
            return;
        }

        // Task 8: Read the contents of "characterExample.txt" using character streams and display them on the console
        try (Reader reader = new FileReader(characterFileName)) {
            StringBuilder contentBuilder = new StringBuilder();
            int character;
            while ((character = reader.read()) != -1) {
                contentBuilder.append((char) character);
            }
            String content = contentBuilder.toString();
            System.out.println(x:"\n## Contents of the file is:");
            System.out.println(content);
        } catch (IOException e) {
            System.out.println("An error occurred while reading the file: " + e.getMessage());
            return;
        }

        // Task 9: Delete the file "example.txt"
        if (file.delete()) {
            System.out.println("\nFile deleted: " + file.getAbsolutePath());
        } else {
            System.out.println("Failed to delete file: " + file.getAbsolutePath());
        }

        // Task 10: Delete the directory "Demo"
        if (directory.delete()) {
            System.out.println("Directory deleted: " + directory.getAbsolutePath());
        } else {
            System.out.println("Failed to delete directory: " + directory.getAbsolutePath());
```

Output:



```java
            // Task 10: Delete the directory "Demo"
            if (directory.delete()) {
                System.out.println("Directory deleted: " + directory.getAbsolutePath());
            } else {
                System.out.println("Failed to delete directory: " + directory.getAbsolutePath());
            }
        }
    }
```

```
cd "/Users/im_zshan/Documents/Labs/Java Lab/Assignment/" && javac FileHandling_Extended.java && java FileHandling_Extended
→  Assignment cd "/Users/im_zshan/Documents/Labs/Java Lab/Assignment/" && javac FileHandling_Extended.java && java FileHandling_Extended
Directory created: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo
File created: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo/example.txt

$ Content written to the file!

## Contents of the file is:
Hi there, I am Zeeshan Chowdhary giving console the very first content.

$ Content written to the file!

## Contents of the file is:
This is an example of writing text using byte streams.

$ Content written to the file!

## Contents of the file is:
This is an example of writing text using character streams.

File deleted: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo/example.txt
Failed to delete directory: /Users/im_zshan/Documents/Labs/Java Lab/Assignment/Demo
→  Assignment
```

Report Submitted to Dr. Jasra Bhat

— The End —