Java File Handling for Beginners

File handling in Java allows us to **create, read, write, and delete files**. It is useful for storing data permanently outside the program. In Java, file handling is mainly done using classes from the `java.io` package like File, FileWriter, FileReader, and Scanner.

1. Creating a File ■

We use the File class and its method createNewFile() to create a new file.

```
import java.io.File;
import java.io.IOException;

public class CreateFileExample {
    public static void main(String[] args) {
        try {
            File myFile = new File("example.txt");
            if (myFile.createNewFile()) {
                System.out.println("File created: " + myFile.getName());
            } else {
                System.out.println("File already exists.");
            }
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

2. Writing to a File ✍■

We use the FileWriter class to write data into a file.

```
import java.io.FileWriter;
import java.io.IOException;

public class WriteFileExample {
    public static void main(String[] args) {
        try {
            FileWriter writer = new FileWriter("example.txt");
            writer.write("Hello, bro! Learning Java File Handling ■");
            writer.close();
            System.out.println("Successfully wrote to the file.");
        } catch (IOException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

3. Reading from a File ■

We can use Scanner or FileReader. Here is an example using Scanner:

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class ReadFileExample {
    public static void main(String[] args) {
        try {
            File myFile = new File("example.txt");
            Scanner reader = new Scanner(myFile);
            while (reader.hasNextLine()) {
                String data = reader.nextLine();
                System.out.println(data);
            }
            reader.close();
        } catch (FileNotFoundException e) {
            System.out.println("An error occurred.");
            e.printStackTrace();
        }
    }
}
```

4. Deleting a File ■

We use the delete() method of File class.

```
import java.io.File;

public class DeleteFileExample {
    public static void main(String[] args) {
        File myFile = new File("example.txt");
        if (myFile.delete()) {
            System.out.println("Deleted the file: " + myFile.getName());
        } else {
            System.out.println("Failed to delete the file.");
        }
    }
}
```

5. Common Exceptions ■■

- IOException → General input/output errors. - FileNotFoundException → File does not exist when trying to read it. - Always remember to close files using close() method to free system resources.

■ Now you know how to create, write, read, and delete files in Java!