

Advance Programming - CPIT305

# HOMEWORK\_1

Shehab

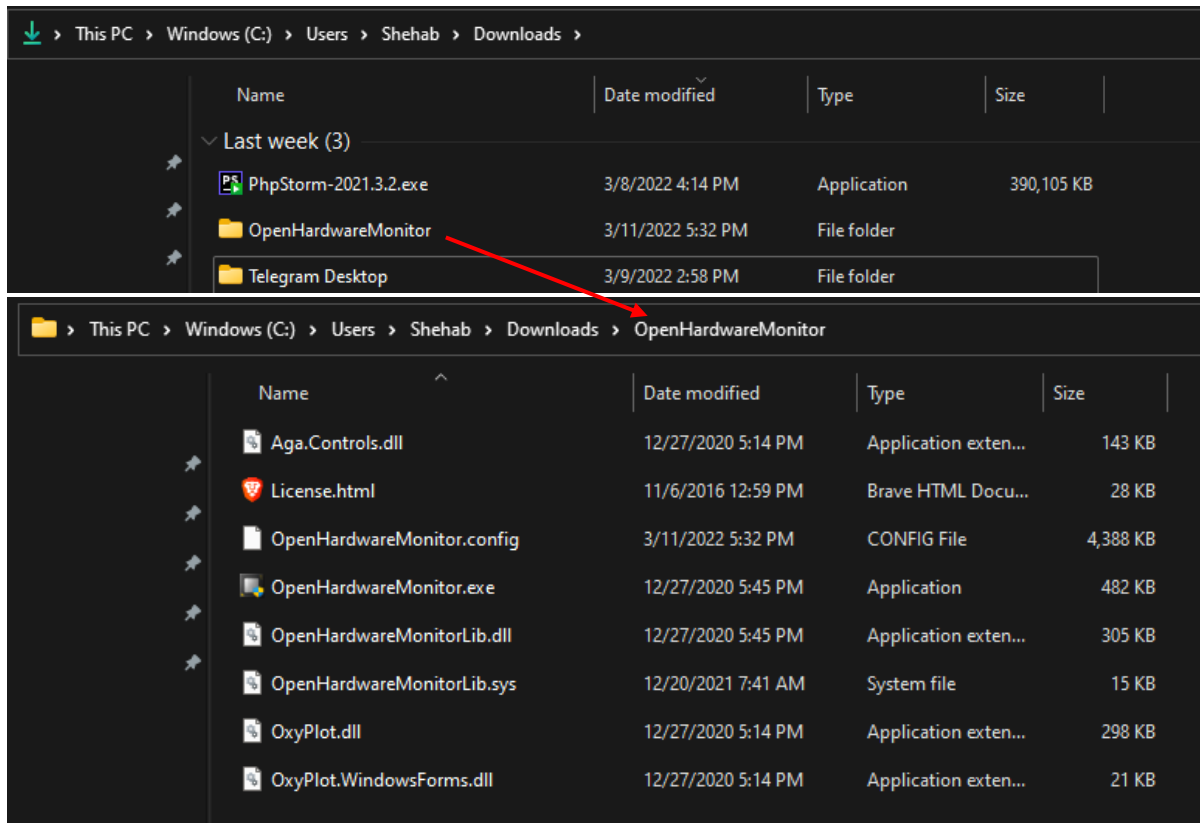
---

## CONTENTS

1-	Main Class .....	2
2-	FileData Class .....	3
1-	File object .....	4
2-	BasicFileAttributes Object .....	4
3-	MD5Checksum Class.....	5
4-	Output.....	6
5-	Refrence.....	7

## 1- MAIN CLASS

```
package com.CodeWithShehab;  
  
public class Main {  
  
    public static void main(String[] args) {  
        new FileData("C:\\Users\\Shehab\\Downloads\\OpenHardwareMonitor").getData();  
    }  
}
```



Main class with the files that the program will execute on.

**Note:** Project has been created using **IntelliJ idea** not **NetBeans** so it might not be executable on **NetBeans**.

## 1- FILEDATA CLASS

```
package com.CodeWithShehab;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.nio.file.attribute.BasicFileAttributes;

public class FileData {
    private File path;
    private BasicFileAttributes attr;

    public FileData(String path) {
        this.path = new File(path);
        if (!this.path.exists()) {
            try {
                throw new FileNotFoundException("File Not Found!");
            } catch (FileNotFoundException e) {
                System.err.println(e.getMessage());
                // so the program stops
                System.exit(0);
            }
        }
        try {
            attr = Files.readAttributes(Paths.get(this.path.getAbsolutePath()),
BasicFileAttributes.class);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public void getData() {
        PrintWriter print = null;
        try {
            print = new PrintWriter(new File("output.txt"));
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        if (path.isFile()) {
            print.println("> File name is           : " + path.getName());
            print.println("> File type           : File");
            print.println("> Created in           : " + attr.creationTime());
            print.println("> Last accessed in      : " + attr.lastAccessTime());
            print.println("> Last modified in     : " + attr.lastModifiedTime());
            print.println("> Hash using MD5 algorithm : " +
MD5Checksum.getMd5Checksum(path.getAbsolutePath()));
            try {
                print.println("> Size           : " +
(Files.size((path.toPath())) * 0.001 + " KB");
            } catch (IOException e) {
                e.printStackTrace();
            }
            print.println("> is File readable?           : " + path.canRead());
            print.println("> is File writable?          : " + path.canWrite());
            print.println("> is File executable?       : " + path.canWrite());
            print.println("*****");
        } else {
            File[] fileArray = path.listFiles();
            for (int i = 0; i < fileArray.length; i++) {
                String fileType = "";
                if (fileArray[i].isDirectory())
                    fileType = "Directory".toUpperCase();
                else
                    fileType = "File".toUpperCase();
                print.println("> File name is           : " + fileArray[i].getName());
                print.println("> File type           : " + fileType);
                print.println("> Created in           : " + attr.creationTime());
            }
        }
    }
}
```

```

        print.println("> Last accessed in          : " + attr.lastAccessTime());
        print.println("> Last modified in          : " + attr.lastModifiedTime());
        try {
            print.println("> Size                      : " +
                (Files.size((path.toPath())) * 0.001 + " KB");
        } catch (IOException e) {
            e.printStackTrace();
        }
        print.println("> Hash using MD5 algorithm : " +
            MD5Checksum.getMd5Checksum(fileArray[i].getAbsolutePath()));
        print.println("> is File readable?          " + fileArray[i].canRead());
        print.println("> is File writable?          " + fileArray[i].canWrite());
        print.println("> is File executable?        " + fileArray[i].canExecute());
        print.println("*****");
    }
    }
    print.close();
}

attr.lastModifiedTime();
try {
    System.out.println("> Size                      : " +
        (Files.size((path.toPath())) * 0.001 + " KB");
} catch (IOException e) {
    e.printStackTrace();
}

    System.out.println("> Hash using MD5 algorithm : " +
        MD5Checksum.getMd5Checksum(fileArray[i].getAbsolutePath()));
    System.out.println("> is File readable?          " +
        fileArray[i].canRead());
    System.out.println("> is File writable?          " +
        fileArray[i].canWrite());
    System.out.println("> is File executable?        " +
        fileArray[i].canExecute());
    System.out.println("*****");
}
}
}

```

This class has 2 fields

## 1- FILE OBJECT

To store the path

## 2- BASICFILEATTRIBUTES OBJECT

Because this class can bring (Creation time, Last access time, Last modified time) of the file/directory. The output for each file will be at the [output section](#) at the end document.

getData() -> One method only to bring the data.

### 3- MD5CHECHSUM CLASS

```
package com.CodeWithShehab;

import java.io.*;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class MD5Checksum {
    public static byte[] createChecksum(String filename) {
        InputStream fis = null;
        try {
            fis = new FileInputStream(filename);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        byte[] buffer = new byte[1024];
        MessageDigest complete = null;
        try {
            complete = MessageDigest.getInstance("MD5");
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
        int numRead = 0;

        do {
            try {
                numRead = fis.read(buffer);
            } catch (IOException e) {
                e.printStackTrace();
            }
            if (numRead > 0) {
                complete.update(buffer, 0, numRead);
            }
        } while (numRead != -1);

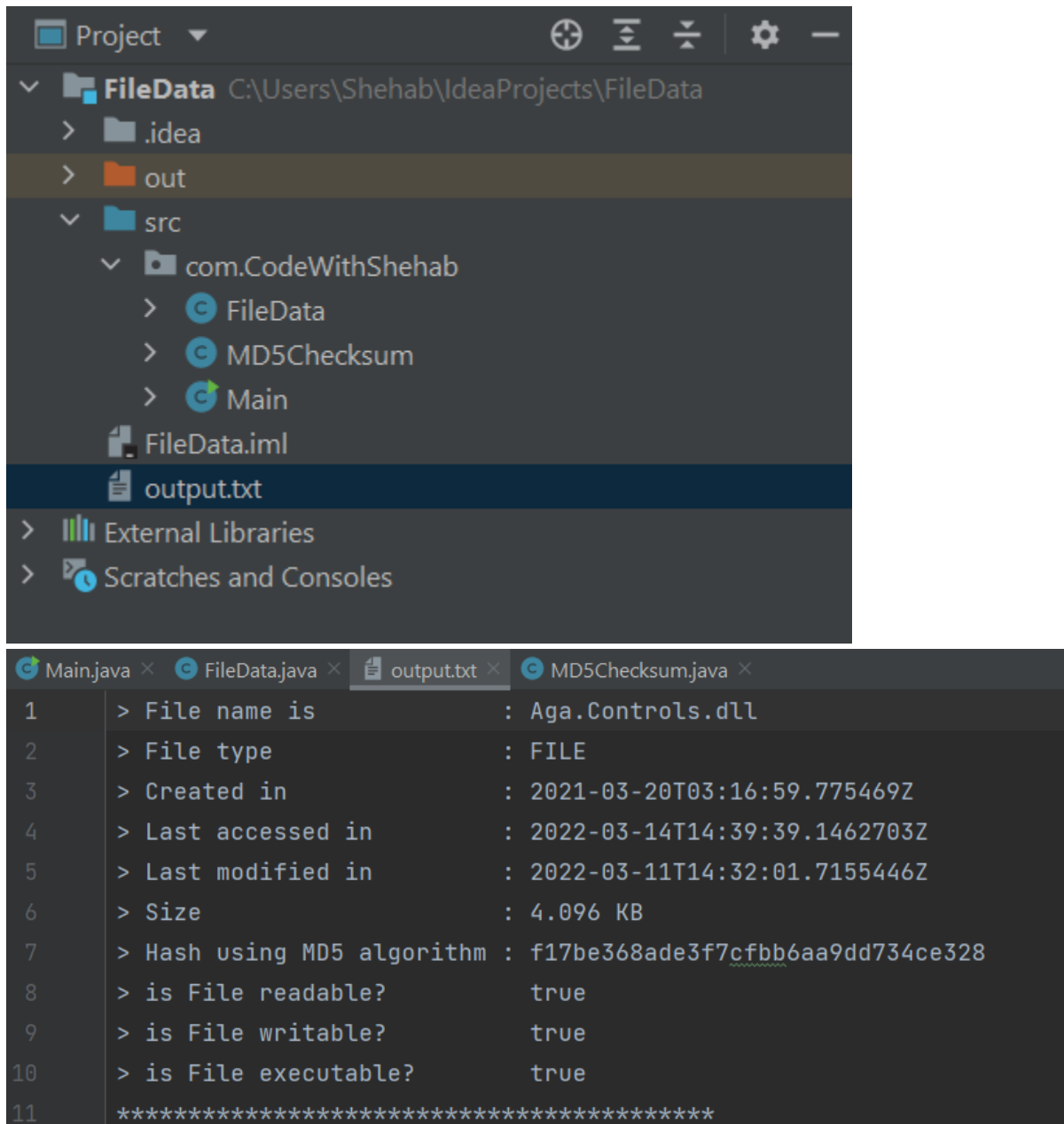
        try {
            fis.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return complete.digest();
    }

    // see this How-to for a faster way to convert
    // a byte array to a HEX string
    public static String getMD5Checksum(String filename) {
        byte[] b = createChecksum(filename);
        String result = "";

        for (int i=0; i < b.length; i++) {
            result += Integer.toString( ( b[i] & 0xff ) + 0x100, 16).substring( 1 );
        }
        return result;
    }
}
```

This class I've found it on [Stackoverflow](#) website it does calculate the hash using MD5 algorithm, I haven't change much in it I just managed the exceptions.

#### 4- OUTPUT



The screenshot displays an IDE interface. The top panel shows the project structure for 'FileData' located at 'C:\Users\Shehab\IdeaProjects\FileData'. The 'out' directory is highlighted. Below it, the 'src' directory contains a package 'com.CodeWithShehab' with classes 'FileData', 'MD5Checksum', and 'Main'. A file 'FileData.iml' is also present. The bottom panel shows the 'output.txt' file, which contains the following text:

```
1 > File name is : Aga.Controls.dll
2 > File type : FILE
3 > Created in : 2021-03-20T03:16:59.775469Z
4 > Last accessed in : 2022-03-14T14:39:39.1462703Z
5 > Last modified in : 2022-03-11T14:32:01.7155446Z
6 > Size : 4.096 KB
7 > Hash using MD5 algorithm : f17be368ade3f7cfbb6aa9dd734ce328
8 > is File readable? true
9 > is File writable? true
10 > is File executable? true
11 *****
```

The output will be saved in a text file inside the project using PrintWriter Object.

Thanks.

## 5- REFERENCE

- 1- <https://stackoverflow.com/questions/304268/getting-a-files-md5-checksum-in-java> Accessed date : 13/3/2022