Java - Files

Md. Mohsin Uddin

East West University

mmuddin@ewubd.edu

July 13, 2019

Byte Streams

```
import java.io.*;
public class CopyFile {
    public static void main(String args[]) throws IOException {
        FileInputStream in = null;
        FileOutputStream out = null;
        try {
            in = new FileInputStream("input.txt");
            out = new FileOutputStream("output.txt");
            int c;
            while ((c = in.read()) != -1) {
                out.write(c);
            }
        }finally {
            if (in != null) { in.close(); }
            if (out != null) { out.close(); }
        }
    }
}
```

Character Streams

```
import java.io.*;
public class CopyFile {
    public static void main(String args[]) throws IOException {
        FileReader in = null;
        FileWriter out = null;
        try {
            in = new FileReader("input.txt");
            out = new FileWriter("output.txt");
            int c;
            while ((c = in.read()) != -1) {
                out.write(c);
            }
        }finally {
            if (in != null) { in.close(); }
            if (out != null) { out.close(); }
        }
    }
}
```

User defined Utility class: Part I

```
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileWriter;
import java.io.IOException;
import java.io.PrintWriter;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import java.util.Scanner;
import java.util.stream.Collectors;

public class Utils {
```

User defined Utility class: Part II

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

}

public static ArrayList<String> getLineArray(String filename) {
    ArrayList<String> tempList = new ArrayList<>();
    File file = new File(filename);
    try {
        Scanner scanner = new Scanner(file);
        while (scanner.hasNextLine()) {
            tempList.add(scanner.nextLine());
        }
        scanner.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    return tempList;
}
```

User defined Utility class: Part III

```
public static void printToFile(String value,
        String filePath, boolean isAppend)
        throws IOException {
    try (PrintWriter out = new PrintWriter(
            new BufferedWriter(
                    new FileWriter(filePath, isAppend)))) {
        out.println(value);
        out.close();
    }
}

public static void storeData(String outputFileLocation,
        String outputText, boolean append) {
    try {
        File file = new File(outputFileLocation);
        if (!file.exists()) {
            file.createNewFile();
        }
```

User defined Utility class: Part IV

```
            FileWriter fileWritter =
                    new FileWriter(outputFileLocation, append);
            try (BufferedWriter bufferWritter =
                    new BufferedWriter(fileWritter)) {
                bufferWritter.write(outputText);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
}

public static List<File> getAllFilesInFolder(String path)
            throws IOException {
        List<File> filesInFolder=Files.walk(Paths.get(path))
                                .filter(Files::isRegularFile)
                                .map(Path::toFile)
                                .collect(Collectors.toList());
        return filesInFolder;
}
```

User defined Utility class: Part V

```
public static List<File> getAllFilesInFolder(File directory)
        throws IOException {
    String path= directory.getAbsolutePath();
    List<File> filesInFolder = Files.walk(Paths.get(path))
                               .filter(Files::isRegularFile)
                               .map(Path::toFile)
                               .collect(Collectors.toList());
    return filesInFolder;
}

public static List<Path> getAllFilePathsInFolder(String path)
        throws IOException {
    List<Path> filesInFolder=Files.walk(Paths.get(path))
                             .filter(Files::isRegularFile)
                             .collect(Collectors.toList());
    return filesInFolder;
}
```

User defined Utility class: Part VI

```
public static List<File> listf(String directoryName) {
    File directory = new File(directoryName);
    List<File> resultList = new ArrayList<>();

    // get all the files from a directory
    File[] fList = directory.listFiles();
    resultList.addAll(Arrays.asList(fList));
    for (File file : fList) {
        if (file.isFile()) {
            System.out.println(file.getAbsolutePath());
        } else if (file.isDirectory()) {
            resultList.addAll(listf(file.getAbsolutePath()));
        }
    }
    return resultList;
}

public static String getCurrentWorkingDir() {
    String currentWorkingDir=System.getProperty("user.dir");
```

User defined Utility class: Part VII

```
        return currentWorkingDir;
}


public static void createDirs(String path) {
        File outputDirectory = new File(path);
        boolean success = outputDirectory.mkdirs();
        //boolean success = outputDirectory.mkdir();
        if (!success) {
                System.out.println("Directory creation failed!");
        }
}



}
```

References


DEITEL, Java How to Program, 11/e


Java: the complete reference, Herbert Schildt, McGraw-Hill Education Group