Files and Streams

Using Streams, Files, Serialization

SoftUni Foundation

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

Table of Contents

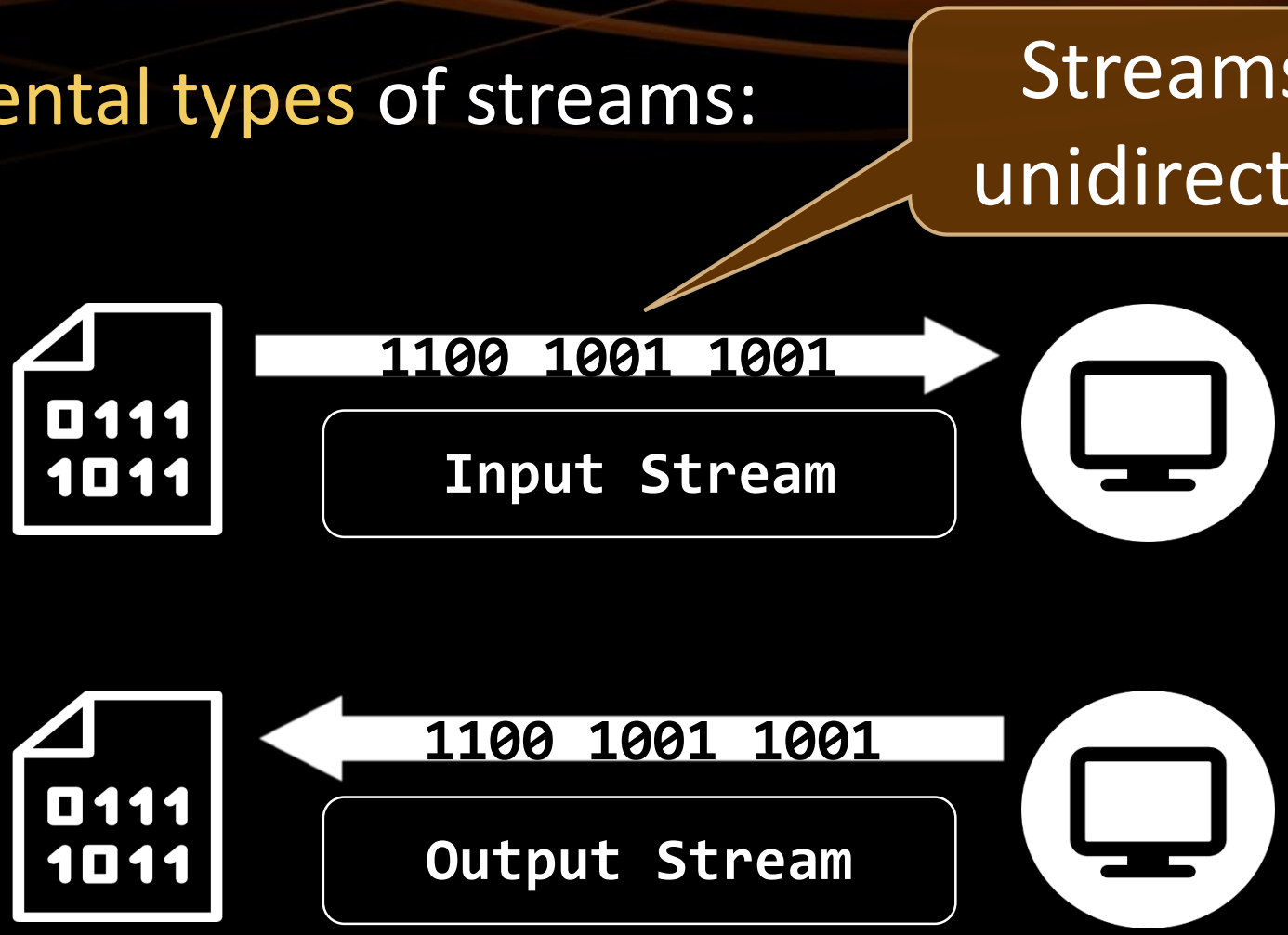
1. Streams Basics
2. Closing a Stream
3. Types of Streams
4. Combining Streams
5. Files and Directories
6. Serialization

What is Stream?

- Streams are used to transfer data
- We open a stream to:
  - Read a file
  - Write to a file

Streams Basics

- Two fundamental types of streams:



Opening a File Stream

SoftUni Foundation

```
String path = "C:\\input.txt";

FileInputStream fileStream =
            new FileInputStream(path);

int oneByte = fileStream.read();
while (oneByte >= 0) {
  System.out.print(oneByte);
  oneByte = fileStream.read();
}
```

Returns -1 if empty

Closing a Stream

- Using try-catch-finally

```
try {
   InputStream in = new FileInputStream(path)
} catch (IOException e) {
   // TODO: handle exception
} finally {
   if (in != null) {
      in.close();
   }
}
```

close() can also throw an exception

Always free resources!

Closing a Stream (2)

SoftUni Foundation

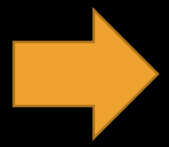
- Using try-with-resources

```
try (InputStream in = new FileInputStream(path)) {
  int oneByte = fileStream.read();
  while (oneByte >= 0) {
    System.out.print(oneByte);
    oneByte = fileStream.read();
  }
} catch (IOException e) {
  // TODO: handle exception
}
```

Problem: Read File

- You are given a file
- Read and print all of its contents as a sequence of bytes
- Submit in Judge only the output of the program

```
Two households, both
alike in dignity,
In fair Verona, where we
lay our scene,
```



```
1010100 1110111 1101111
100000 1101000 1101111
1110101 1110011 1100101
1101000…
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/403#0

Solution: Read File

SoftUni Foundation

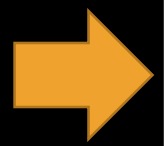
```
String path = "D:\\input.txt";

try (InputStream in = new FileInputStream(path)) {
  int oneByte = in.read();
  while (oneByte >= 0) {
    System.out.prinf("%s ",
        Integer.toBinaryString(oneByte));
    oneByte = in.read();
  }
}
catch (IOException e) {
  e.printStackTrace();
}
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#0

Problem: Write to File

- Read a file and write all its content while skipping any punctuation (skip ',', '.', '!', '?')
- Submit in Judge only the output of the program

```
Two households, both
alike in dignity.
In fair Verona, where
we lay our scene.
```



```
Two households both
alike in dignity
In fair Verona where
we lay our scene
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#1

Solution: Write to File

SoftUni Foundation

```
String inputPath = "D:\\input.txt";
String outputPath = "D:\\output.txt";

List<Character> symbols = new ArrayList<>();
Collections.addAll(symbols, '.', ',', '!', '?');

// continues…
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#1

Solution: Write to a File

SoftUni Foundation

```
try (InputStream in = new FileInputStream(inputPath);
     OutputStream out = new FileOutputStream(outputPath))
{
  int oneByte = 0;
  while ((oneByte = in.read()) >= 0) {
    if (!symbols.contains((char)oneByte)) {
      out.write(oneByte);
    }
  }
} // TODO: handle exceptions
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#1

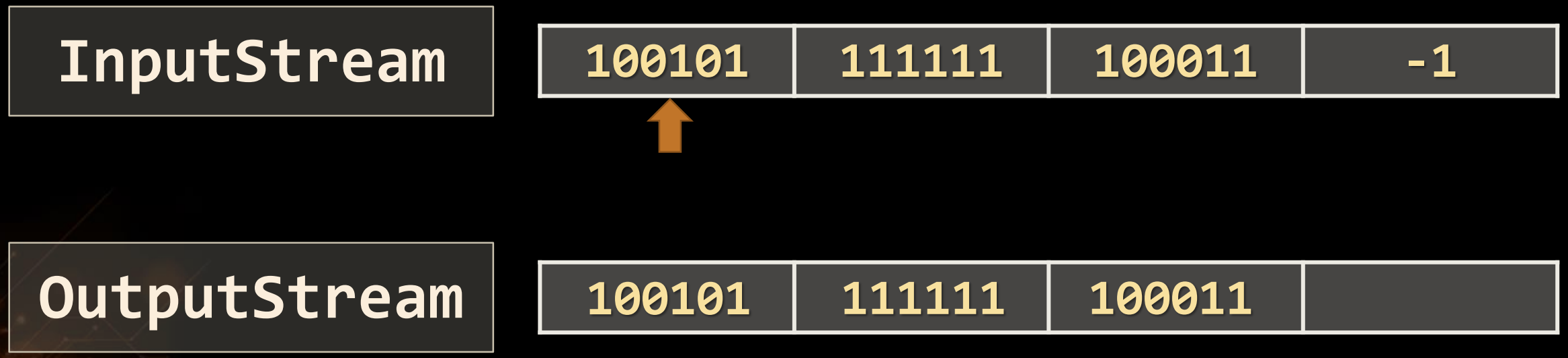
SoftUni Foundation

Basic Stream Types in Java

Byte, Character

Byte Stream

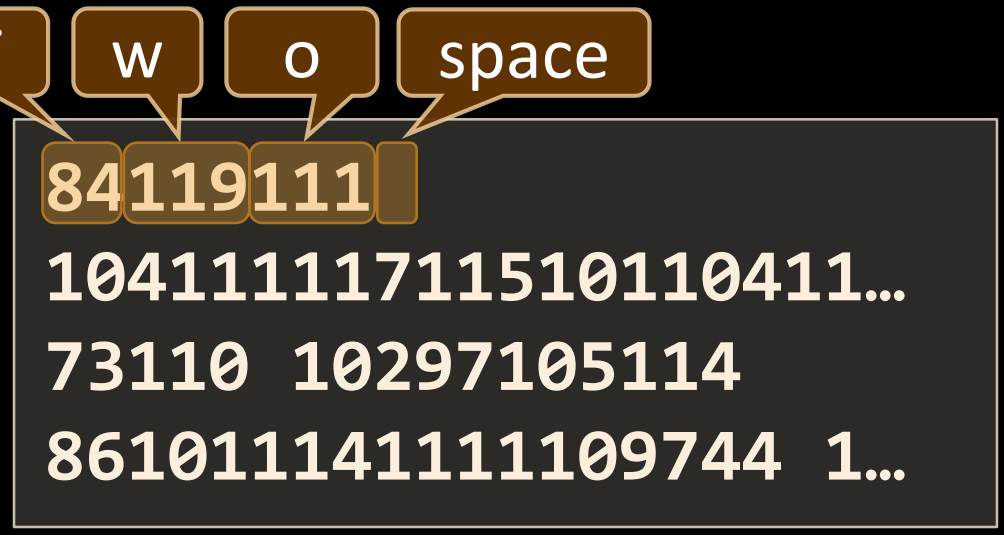
- Byte streams are the lowest level streams
  - Byte streams can read or write one byte at a time
  - All byte streams descend from InputStream and OutputStream



Problem: Copy Bytes

- Read a file and copy its contents to another text file
- Write characters as bytes in decimal
- Write every space or new line as it is, e.g. as a space or new line

```
Two households, both
alike in dignity.
In fair Verona, where
we lay our scene.
```



Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#2

Solution: Copy Bytes

SoftUni Foundation

```
int oneByte = 0;
while ((oneByte = in.read()) >= 0) {
  if (oneByte == 10 || oneByte == 32) {
    out.write(oneByte);
  } else {
    String digits = String.valueOf(oneByte);
    for (int i = 0; i < digits.length(); i++)
      out.write(digits.charAt(i));
  }
}
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#2

Character Streams

SoftUni Foundation

- All character streams descend from `FileReader` and `FileWriter`

```
String path = "D:\\input.txt";

FileReader reader = new FileReader(path);
```

Combining Streams

SoftUni Foundation

- Character streams are often "wrappers" for byte streams.
  - `FileReader` uses `FileInputStream`
  - `FileWriter` uses `FileOutputStream`

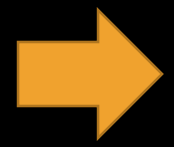
```
String path = "D:\\input.txt";

Scanner reader =
  new Scanner(new FileInputStream(path));
```

Wrapping a Stream

Problem: Extract Integers

- Read a file and extracts all integers in a separate file
- Get only numbers that are not a part of a word
- Submit in Judge only the output of the program

```
2 households, 22 alike
in 3nity,
In fair Verona, where
we lay our scene
```



```
2
22
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#3

Solution: Extract Integers

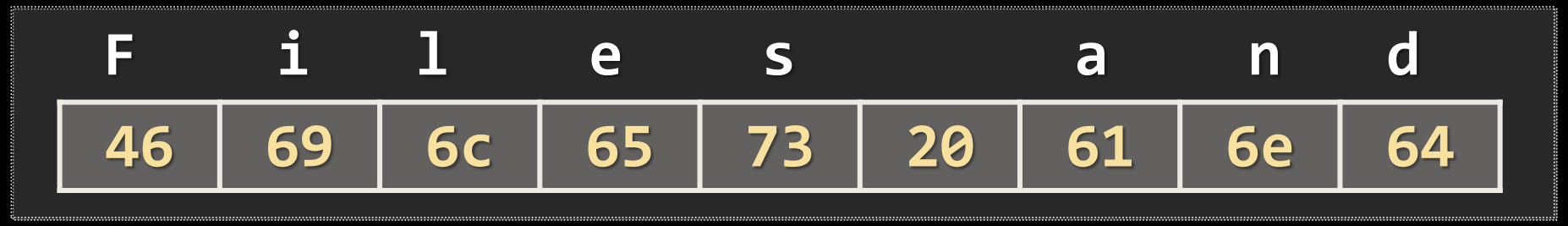
SoftUni Foundation

```
Scanner scanner =
      new Scanner(new FileInputStream(inputPath));

PrintWriter out =
      new PrintWriter(new FileOutputStream(outputPath))) {

while (scanner.hasNext()) {
  if (scanner.hasNextInt())
    out.println(scanner.nextInt());

  scanner.next();
}
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#3

Buffered Streams

- Reading information in chunks
- Significantly boost performance



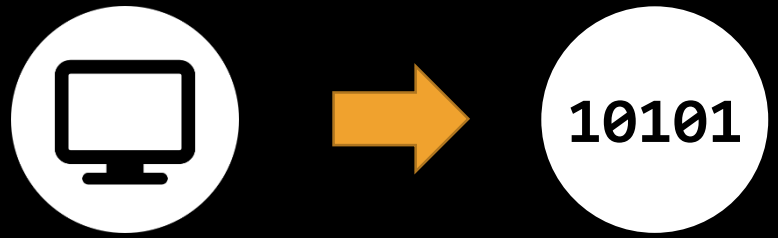
Problem: Write Every Third Line

SoftUni Foundation

- Read a file and write all lines which number is divisible by 3 in a separate file
- Line numbers start from one

```
Two households, both
alike in dignity,
In fair Verona, where
we lay our scene,
From ancient grudge
break to new mutiny…
```

Lines start at 1



Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#4

Solution: Write Every Third Line

SoftUni Foundation

```
try (BufferedReader in =
        new BufferedReader(new FileReader(inputPath));
     PrintWriter out =
        new PrintWriter(new FileWriter(outputPath))) {
  int counter = 1;
  String line = in.readLine();
  while (line != null) {
    if (counter % 3 == 0)
      out.println(line);
    counter++;
    line = in.readLine();
  }
} // Catch Exception
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#4

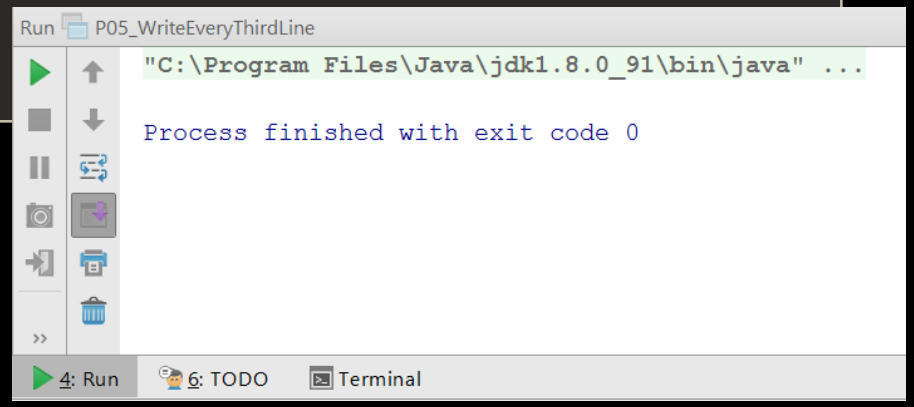
Command Line I/O

- Standard Input – `System.in`
- Standard Output – `System.out`
- Standard Error – `System.err`

```
Scanner scanner = new Scanner(System.in);
String line = scanner.nextLine();
System.out.println(line);
```

Input Stream

Output Stream



SoftUni Foundation

Basic Stream Types

Live Exercises in Class (Lab)

Files and Paths

Easily Working With Files

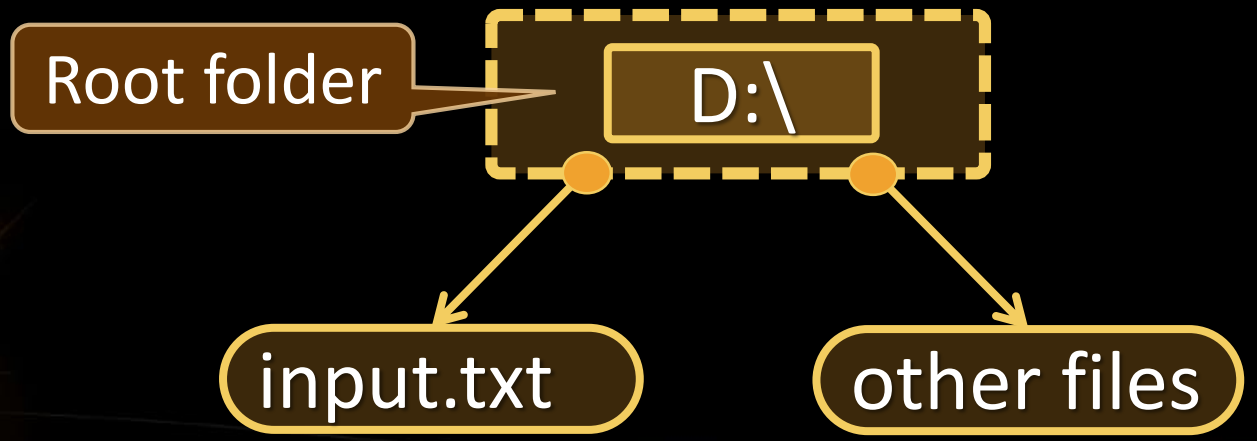
Paths

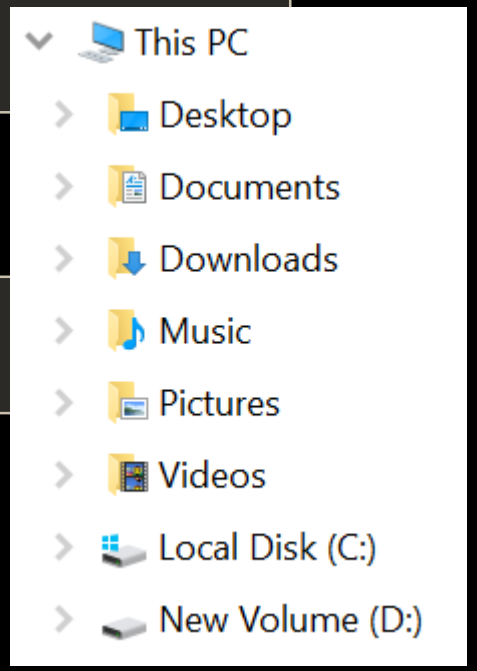
- The location of a file in the file system

```
D:\input.txt
```

- Represented in Java by the Path class

```
Path path = Paths.get("D:\\input.txt");
```





Files

- Provides static methods for creating streams

```
Path path = Paths.get("D:\\input.txt");

try (BufferedReader reader =
      Files.newBufferedReader(path)) {
  // TODO: work with file
} catch (IOException e) {
  // TODO: handle exception
}
```

Files (2)

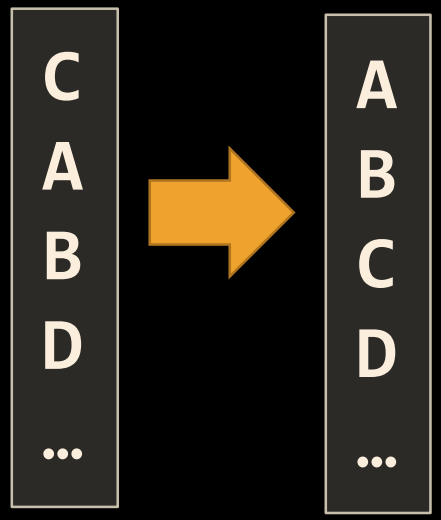
SoftUni Foundation

- Provides utility methods for easy file manipulation

```
Path inPath = Paths.get("D:\\input.txt");
Path outPath = Paths.get("D:\\output.txt");

List<String> lines = Files.readAllLines(inPath);
Files.write(outPath, lines);

// TODO: handle exceptions
```

Problem: Sort Lines

- Read a text file and sort all lines
- Write the result to another text file
- Use Paths and Files classes



Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#5

Solution: Sort Lines

SoftUni Foundation

```
Path path = Paths.get("D:\\input.txt");
Path output = Paths.get("D:\\output.txt");

try {
  List<String> lines = Files.readAllLines(path);
  Collections.sort(lines);
  Files.write(output, lines);
} catch (IOException e) {
  e.printStackTrace();
}
```

Don't use for large files

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#5

SoftUni Foundation

File Class in Java

Easily Working With Files

File Class in Java

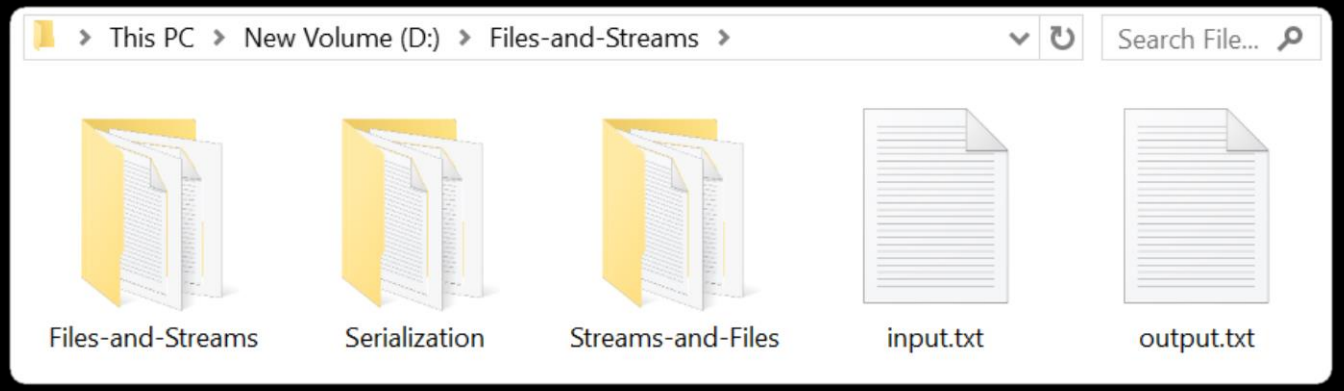
SoftUni Foundation

- Provides methods for quick and easy manipulation of files

```
import java.io.File;

File file = new File("D:\\input.txt");

boolean isExisting = file.exists();
long length = file.length();
boolean isDirectory = file.isDirectory();
File[] files = file.listFiles();
```

Problem: List Files

- Print names and sizes of all files in "Files-and-Streams" directory
- Skip child directories




```
input.txt: [size in bytes]
output.txt: [size in bytes]
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#6

Solution: List Files

SoftUni Foundation

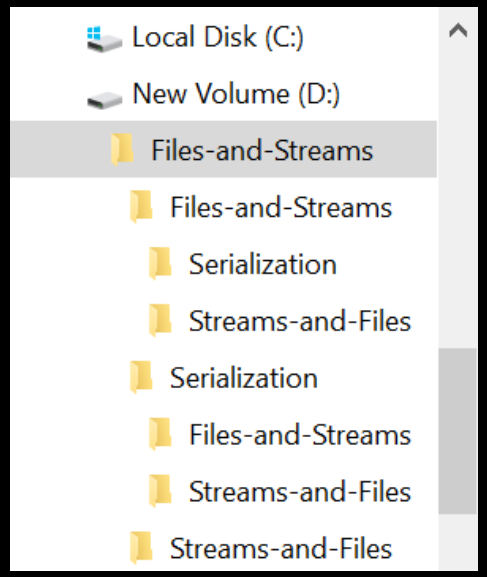
```
if (file.exists())
  if (file.isDirectory())
    File[] files = file.listFiles();
    for (File f : files)
      if (!f.isDirectory())
        System.out.println(f.length());
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#6

Problem: List Nested Folders

SoftUni Foundation

- You are given a folder named "Files-and-Streams"
- List all folder names, starting with the root
- Print folder count on the last line (including the root)




```
…
Streams-and-Files
Serialization
Streams-and-Files
[count] folders
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#7

Solution: Nested Folders

SoftUni Foundation

```
String path = "D:\\Files-and-Streams";
File root = new File(path);

Deque<File> dirs = new ArrayDeque<>();
dirs.offer(root);

// continue…
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#7

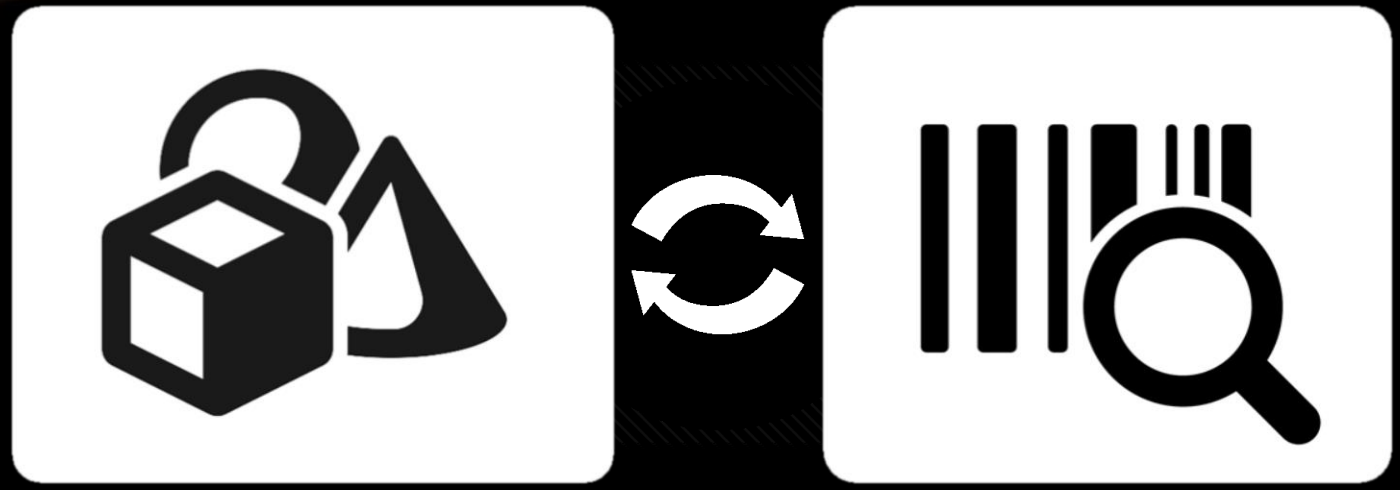
Solution: Nested Folders (2)

SoftUni Foundation

```
int count = 0;
while (!dirs.isEmpty()) {
        File current = dirs.poll();
        File[] nestedFiles = current.listFiles();
        for (File nestedFile : nestedFiles)
                if (nestedFile.isDirectory())
                        dirs.offer(nestedFile);
                        count++;
        System.out.println(current.getName());
}
System.out.println(count + " folders");
```

Check your solution here: https://judge.softuni.bg/Contests/Practice/Index/1032#7

SoftUni Foundation



Serialization

Serializing and Deserializing Objects

Serialization

- Save objects to a file

Save objects to .ser file

```
List<String> names = new ArrayList<>();
Collections.addAll(names, "Mimi", "Gosho");

FileOutputStream fos = new FileOutputStream(path);
ObjectOutputStream oos =
        new ObjectOutputStream(fos);

oos.writeObject(names);

// TODO: handle exceptions
```

Deserialization

- Load objects from a file

```
FileInputStream fis =
        new FileOutputStream(path);
ObjectInputStream oos =
        new ObjectInputStream(fis);

List<String> names =
        (List<String>) oos.readObject();

// TODO: handle exceptions
```

Serialization of Custom Objects

SoftUni Foundation

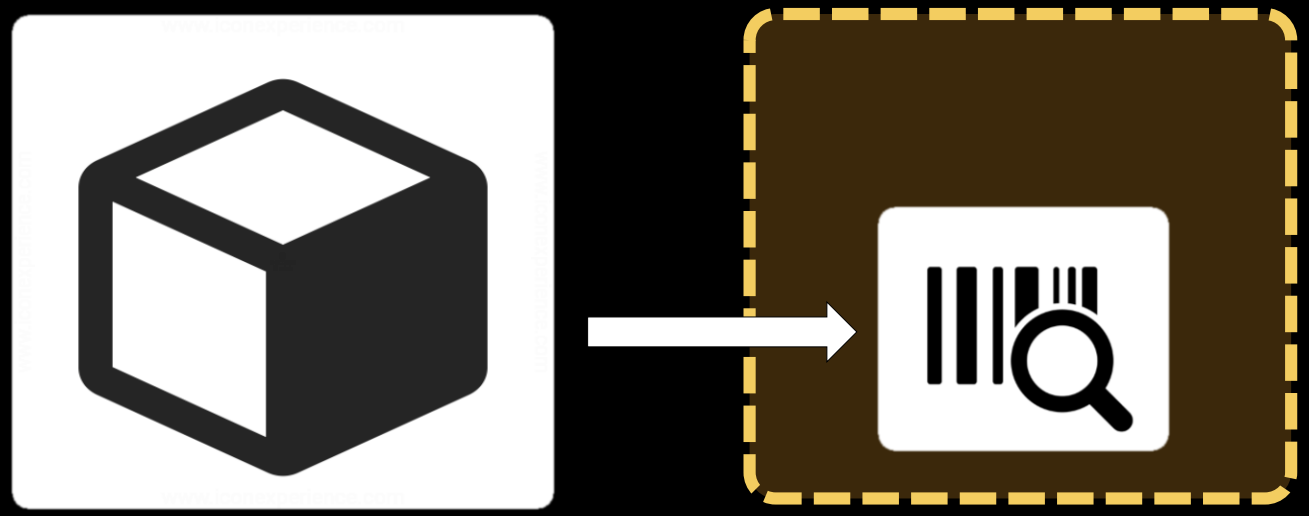
- Custom objects should implement the Serializable interface

```
class Cube implements Serializable {
  String color;
  double width;
  double height;
  double depth;
}
```

Problem: Serialize Custom Object

SoftUni Foundation

- Create a Cube class with color, width, height and depth
- Create a cube – color: "green", w: 15.3, h: 12.4 and d: 3



Solution: Serialize Custom Object

SoftUni Foundation

```
class Cube implements Serializable {
  String color;
  double width;
  double height;
  double depth;
}
```

Solution: Serialize Custom Object (2)

SoftUni Foundation

```
String path = "D:\\save.ser";
try (ObjectOutputStream oos =
        new ObjectOutputStream(
          new FileOutputStream(path))) {
   oos.writeObject(cube);
} catch (IOException e) {
   e.printStackTrace();
}
```

SoftUni Foundation

Files and Serialization

Live Exercises in Class (Lab)

Summary

- Streams are used to transfer data
- Two main types of streams
  - Input Streams
  - Output Streams
- Buffered streams boost performance
- Streams can be chained together
- You can save objects state into a file

Java Advanced – Course Overview

SoftUni Foundation

Questions?

https://softuni.bg/courses/java-advanced

License

SoftUni Foundation

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

- Attribution: this work may contain portions from
  - "Fundamentals of Computer Programming with Java" book by Svetlin Nakov & Co. under CC-BY-SA license

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

Software University

SoftUni Foundation

FORUM