Week9 – Error Handling and File I/O

COMP90041 Programming and software development

Prepared By: Chenyang Lu

THE UNIVERSITY OF
MELBOURNE

Exception Handling

Try-throw-catch block

```
try {
  //  Block of code to try
}
catch(Exception e) {
  //  Block of code to handle errors
}
```

Exception Class

Exception is the parent class of all exception classes, we have many other exception classes:

- IOException
- ClassNotFoundException

You can also define your own Exception class

THE UNIVERSITY OF
MELBOURNE

Text File I/O

Write a Text File (ASCII File)

Step1: Create FileOutputStream Object (use filename as a parameter):

```
FileOutputSrteam  outputFile= new FileOutputSrteam(FileName)
```

Step2: Create PrintWriter Object (use FileoutputStream as a parameter):

```
PrintWriter outPutStream = new PrintWriter(outputFile)
```

Step3: invoke methods of PrintWriter Class:

```
outPutStream.print()
outputStream.println()
flush()
close()
```

Easy way: `PrintWriter outPutStream = new PrintWriter(new FileOutputSrteam(FileName))`

Write a Text File (ASCII File) --- Second Approach

Step1: Create FileOutputStream Object (use filename as a parameter):

FileOutputSrteam outputFile= new FileOutputSrteam(FileName)

Step2: Create PrintWriter Object (use FileoutputStream as a parameter):

PrintWriter outPutStream = new PrintWriter(outputFile)

Step3: invoke methods of PrintWriter Class:

outPutStream.print()

outputStream.println()

flush()

close()

Easy way: PrintWriter outPutStream = new PrintWriter(new FileOutputSrteam(FileName))

Read Text File

Step1: Create FileINStream Object (use filename as a parameter):

FileInputSrteam inputFile= new FileInputSrteam(FileName)

Step2: Create Scanner Object (use FileoutputStream as a parameter):

Scanner inputStream = new Scanner(inputFile)

Step3: invoke methods of Scanner Class:

inputStream.nextLine()

inputStream.nextInt()

inputStream.nextByte()

intputStream.hasNext()

Read Text File

Step1: Create FileINStream Object (use filename as a parameter):

FileInputSrteam inputFile= new FileInputSrteam(FileName)

Step2: Create Scanner Object (use FileoutputStream as a parameter):

Scanner inputStream = new Scanner(inputFile)

Step3: invoke methods of Scanner Class:

inputStream.nextLine()

inputStream.nextInt()

inputStream.nextByte()

intputStream.hasNext()

Easy way: Scanner inputStream = new Scanner(new FileInputSrteam(FileName)

)

Other approaches to read and write text file

Read: **BufferedReader inputStream= new BufferedReader(new FileReader(*FileName*));**

Methods:

inputStream .readLine()

Other approaches to read and write text file

Read: **BufferedReader inputStream= new BufferedReader(new FileReader(*FileName*));**

Methods:

inputStream .readLine()

THE UNIVERSITY OF
MELBOURNE

Binary File I/O

Write binary File

ObjectOutputStream *outputStreamName* = new ObjectOutputStream(new FileOutputStream(*FileName*));

Methods:
writeInt()
writeDouble()
writeChar()
writeBoolean()
writeObject()
flush()
close()

Read binary File

ObjectInputStream *inputStream* = new ObjectInputStream(new FileInputStream(*FileName*));

Methods:
readInt()
readShort()
readLong()
readDouble()
readObject()

Binary I/O of Objects

- It is best to store the data of only one class type in any one file

-the class of the object being read or written must implement the ***Serializable*** *interface*