Files and Exception Handling

Ho Dac Hung

What is a File?

- A file is a collection of related data stored on a persistent medium such as a hard disk or a CD. Persistent simply means lasting.
- File often store data used by an application. File are also used to store the data generated by an application.

The File Class

- The File class, part of the java.io package, is used for creating an object that represents a file. A File object can be used to create a new file, test for the existence of a file, and delete a file.

  File(String f)

  createNewFile()

  delete()

  exists()

Handling Exception

- An exception is an error affecting program execution. If an exception is not taken care of, or handled, the application abruptly terminates.
- Although many types of exceptios may still require program termination, an exception handler can allow an application to terminate gracefully by providing the user with an informative error message.

Handling Exception

- An exception handler us a block of code that performs an action when an exception occurs. The try-catch-finally statement can be used to write an exception handler.

```
try{
    <statements>
} catch (exception err code){
    <statements>
} finally {
    <statements>
}
```

The File Streams

- A file must be associated with a stream in order to perform operations such as reading the contents, writing over existing contents, and adding to the existing contents. A stream processes characters, and in Java, streams are implementes with classes.

The File Streams

- The file stream keeps track of the file position, which is the point where reading or writing last ocurred. File streams are used to perform sequantial file access, with all the reading and writing perfoemed onc character after another or one line after another.

The FileReader and BufferedReader Classes

- The FileReader amd BufferedReader classed, both from java.io package, are used together to read the contents of an existing file. The FileReader class is used to create an input file stream. Next, the BufferedReader class is used to read text from the stream.

  FileReader(File filename)

  BufferedReader(Reader stream)

  read()

  readLine()

  close()

The FileWriter and BufferedWriter Classes

- The FileWriter and BufferedWriter classes, both from the java.io package, are used together to write data to a file. The FileWriter class is used to create an output file stream. A BufferedWriter class object is then used to send text to the stream.

  FileWriter(File filename, boolean append)

  BufferedWriter(Writer stream)

  newLine()

  write(String str)

  write(char c)

  close()

Object Serialization

- A file can also be used to store object data. Writing objects to a file is called object serialization. In this process, class information about an object is written out to a stream. If a class uses another class, this information is also written out, and so on.

Object Serialization

- Object serialization and deserialization is performed with object output and input stream.
- The FileOutputStream and ObjectOutputStream classes, both from the java.io package, are used together to write objects to a file.

FileOutputStream(File filename, boolean append)

ObjectOutputStream(FileOutputStream stream)

writeObject(Object obj)

writeInt(int num)

writeDouble(double num)

Object Serialization

- TheFileInputStream and ObjectInputStream classes, also from the java.io package, are used together to read objects from a file.

  FileInputStream(File filename)

  ObjectInputStream(FileInputStream stream)

  readObject()

  readInt()

  readDouble()