 

**Advanced Placement Computer Science**

**Unit 6: File Input/Output**

Lesson: Writing and Reading Text Files

*Last Updated:* ***12/2/2012***

Streams are C++ and Java’s way of standardizing input and output

A Stream is a sequence of bytes

A byte is a grouping of 8 bits

Examples of Streams,

Whenever a java application runs, The System static class is created and so is System.in(keyboard) and System.out(screen)

When a text file is written to a local disk, an output stream is created

When a text file is read from a local disk, an input stream is created

The next commands shown below is how Josh Komoroske opened a stream to an internet web server.

URL server = null; //A URL represents a uniform resource location

try {

server = new URL(url); //connect to the target URL

} catch (MalformedURLException m) {

throw new Exception(m);

}

BufferedReader in = null; //a buffered reader can process a stream of information

try {

**in = new BufferedReader(new InputStreamReader(server.openStream())); //open a new stream buffer**

} catch (IOException i) {

throw new Exception(i);

}

There are two different approaches to storing information in files;

1. Text (groups of 8 bits / 16bits) are interpreted as a character
2. Binary: 1’s and 0’s can be interpreted in any way appropriate

There are two different approaches to accessing the stored information in files;

1. Sequential Access: Read from the start to the end
2. Random access: Can skip around the file to read where necessary

Meet the players;

|  |  |  |
| --- | --- | --- |
| Class | Useful for | Example |
| File | **Represents an individual file or directory on a disk system.**  **Used to open and close streams to that file** | //Use the File class to see if this is a directory File searchRoot = new File(fileDir);  if (searchRoot.isDirectory()) |
| FileWriter | **Writing ASCII text to a text file. Often used in conjunction with PrintWriter (see below)** | //Try to open the stream for writing...  try {  FileWriter fw;  fw = new FileWriter("output.txt"); //if file is already there, this will blow it away  PrintWriter pw = new PrintWriter(fw);  pw.println(“Two Roads Diverged in a Wood and I”);  pw.println(“I took the one less traveled by”);  pw.println(“And that has made all the difference”);  }catch(Exception e)  {  System.out.println("Can't open file");  }  fw.close(); |

|  |  |  |
| --- | --- | --- |
| PrintWriter | Allows for easy access to putting information to the file, using print and println.  Works just like System.out except data is written to a text file instead of the console. | See above example |
| FileReader | **Reading ASCII text from an opened text file** | BufferedReader input = new BufferedReader(new FileReader("data.txt")); |
| BufferedReader | **Useful for reading text files one line at a time. (Buffering also helps over a network or internet connection)** | See above |
| StringTokenizer | **Breaks up a String into tokens or smaller Strings** | StringTokenizer st = new StringTokenizer(line, "|"); |
| Scanner | **Scanner can be used to read the text files as well** | try {  //Attempt to open the file  File f = new File(fileName);  //Assuming its open, let's grab the info  Scanner input = new Scanner(f);  while (input.hasNext()) {  //Read the currency  String descript = input.nextLine();  String temp = input.nextLine();  double convertRate = Double.parseDouble(temp);  //Now create a new Currency object  Currency tempCurrency = new Currency(descript, convertRate);  //Add into the array  if (numRates < 100) {  rates[numRates] = tempCurrency;  numRates++;  }  }  input.close();  }  catch (Exception e) {  sc.println(" --------------------------");  sc.println("| FILE problem |");  sc.println(" --------------------------");  sc.println(e); //print the exception  } |

Here is what the rates.txt file looks like;

Euro

0.74895

Great Britain Pounds

0.51481

Japanese Yen

117.650

Chinese Yen

7.74529

Mexican Peso

11.2075

Canadian Dollar

1.17725

Pakistani Rupee

60.7170

NOTE: Be careful with the last line and blank lines at the end of the file!!!

NOTE: Files must be in the project directory (NOT the source or class directory)

Steps for Writing a File:  
1. Open the file for writing (can either APPEND or OVERWRITE)  
APPEND = add to the end of what is there  
  
  
  
  
OVERWRITE = destroy the file

Let’s write out the first and last names from an array to a text file called roster.txt

public static void main(String[] args){

String[ ]names = {“Karl Marx”, “Abe Lincoln”, “Theodore Roosevelt”, “Aung San Suu Kyi”, “Alfred Nobel”};

try {

PrintWriter pw = new PrintWriter(new FileWriter(“roster.txt”));

for(inti =0; i<names.length; i++)

pw.println(names[i]);

pw.close();

} catch (IOException ex) {

}

}

Steps for Reading a File:  
Create a scanner or Buffered Reader

Open the file

Repeat the readln command(or nextLine)

Close the Stream

import java.io.File;

import java.io.FileNotFoundException;

import java.util.ArrayList;

import java.util.Scanner;

import java.util.logging.Level;

import java.util.logging.Logger;

public class ReadNamesUsingScannerSolFromNotes {

public static void main(String[] args) {

ArrayList<String>names = new ArrayList<String>();

try {

Scanner inFile = new Scanner(new File("roster.txt")); //looks in proj folder

while(inFile.hasNext()){

names.add(inFile.nextLine());

}

inFile.close();

} catch (FileNotFoundException ex) {

Logger.getLogger(ReadNamesUsingScannerSolFromNotes.class.getName()).log(Level.SEVERE, null, ex);

}

for (int i = 0; i < names.size(); i++) {

System.out.println(names.get(i));

}

}

}

Mr Hanley’s preferred way of storing information:

I prefer to store information with each record on a single line

As an example, consider this text file

**McCarthy|Walter|255 Grapevine Rd|Wenham|MA|01984|12000.00**

**NaSmith|Courtney|7 Main St.|Clifton Park|NY|12065|18000.00**

**Anderson|Trinity|957 First St.|Hermosa Beach|CA|01954|19000.00**

/\*=============================================

= FILE: ReadFileUnsureSizStringTokSol.java

= DATE: 2/2/2004

= AUTHOR: han1337

= PURPOSE: Demonstrate reading data of unknown size

=============================================\*/

import java.io.\*;

import java.util.\*;

import java.util.StringTokenizer;

/\*File could look like this

McCarthy|Walter|255 Grapevine Rd|Wenham|MA|01984|12000.00

NaSmith|Courtney|7 Main St.|Clifton Park|NY|12065|18000.00

Anderson|Trinity|957 First St.|Hermosa Beach|CA|01954|19000.00

\*/

public class ReadFileUnsureSizeStringTokSol

{

String fname, lname, streetAddr, town, state, zip;

double salary;

public ReadFileUnsureSizeStringTokSol()

{

read();

}

public void read()

{

try

{

BufferedReader input = new BufferedReader(new FileReader("data.txt"));

String line;

//Attempt to read from the file

line = input.readLine();

while (line != null) //goes to the end of file

{

StringTokenizer st = new StringTokenizer(line, "|"); //| is the delimiter

//Now break up the line

lname = st.nextToken();

fname = st.nextToken();

streetAddr = st.nextToken();

town = st.nextToken();

state = st.nextToken();

zip = st.nextToken();

salary = Double.parseDouble(st.nextToken());

System.out.println("Here's our info " + fname + " " + lname + " " +

streetAddr + " " + town + " " + state + " " + zip +

" "

+ salary);

line = input.readLine();

}

input.close();

}

catch (Exception e)

{

System.out.println(e.toString());

}

}

public static void main(String[] args)

{

ReadFileUnsureSizeStringTokSol rfus = new ReadFileUnsureSizeStringTokSol();

}

}