

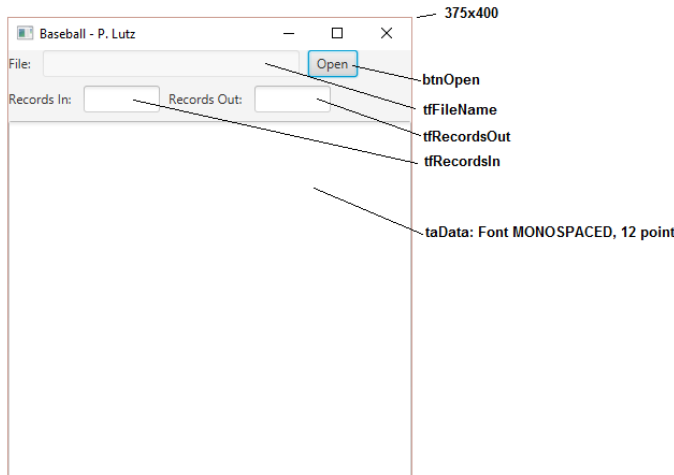
Binary and CSV (Text) I/O

ISTE-121 – Homework 03

Problem Statement:

An extract of some names and personal information has been made from a baseball database. This assignment is to read the data from a comma separated value text file (CSV), write the data into a binary file, then read it back in writing the data to the screen showing that the data in the binary file is intact and correct.

You must implement the following GUI:



When the user clicks the Open button, display a FileChooser that shows CSV files in the current directory and let the user choose one.

You will create a similarly named .dat file. Example: Given filename BaseballNames2.csv, the output file is named BaseballNames2.dat.

Input Datafile layout:

Two CSV datafiles have been provided for testing, BaseballNames1.csv and BaseballNames2.csv. Each contains a header line as the first line of the file. A blank line may or may not follow. Your program must account for a blank or non-blank line (record) regardless of what the sample file shows. The program must also account for leading and trailing spaces or tabs in the data (see the String method **trim** for help with this), along with various errors.

Field name	Data type
Name First	Text
Name Last	Text
Birth Day	Whole number
Birth Month	Whole number
Birth Year	4-digit year
Weight	Whole number
Height	Number with decimal part

BaseballNames1.csv contain no errors, and your program must be able to process this one cleanly before you move to the next datafile.

BaseballNames2.csv may have a value for each expected field and may not. Since this was a database extract, some missing (null) fields show up in the data. Some records may have one or more missing fields.

Write errors to the taData text area and only valid records to the .dat file. When an error is encountered in a line read in from the CSV, write the line and first error found to taData. Then continue to process the rest of the file. Your program will then read your created *.dat data file and print it to taData. Reporting multiple errors per record is a bonus, described below.

Correctly writing the above with Javadocs, can gain you full credit for this homework.

My Solution

Together with the two csv files, the Downloads for HW3 include a jar file for my solution called BaseballA.jar. You can run this to see how your program should run. The bonuses are not provided in this solution.

The program BaseballB.jar is my solution with the bonus levels (both of them) in place.

Bonus levels:

Bonus level 1 is to reporting all errors in any one record. Level 1 allows printing the original record for each error reported. See sample output for Bonus Level 1.

Bonus level 2 is to report all errors, but only display the original record once. **Also**, handle records with not enough fields (7) without a fatal error. See sample output for Bonus Level 2. Level 2 adds to the points from level 1.

Output datafile layout:

The binary output (.dat) file must have the fields in the same order as the input. No headings are written in the output file, only the data. The appropriate binary I/O data types must be used, text must be written as text, numbers must be written as the appropriate numeric data type. You must use the appropriate methods of `DataInputStream` and `DataOutputStream` for all I/O to the ".dat" file.

Output report:

At the end of reading the csv file, and writing the dat file, display how many records were read from the csv file in `tfRecordsIn` and how many records were written to the dat file in `tfRecordsOut`. Remember to count the heading and blank lines as records read in, but not written out. Similarly, records with errors are records read in but not written out.

Use formatted printing, so the output report looks impressive. Spacing does not have to be exactly as shown, but it should be close.

Hints:

Dates are provided in the western format of "month, day, year".

We suggest getting `BaseballNames1.csv` to work first. Save a backup copy of the code, then start on `BaseballNames2.csv`. Make sure you hand in something before the due date. Any bonus points won't make up for the late deductions.

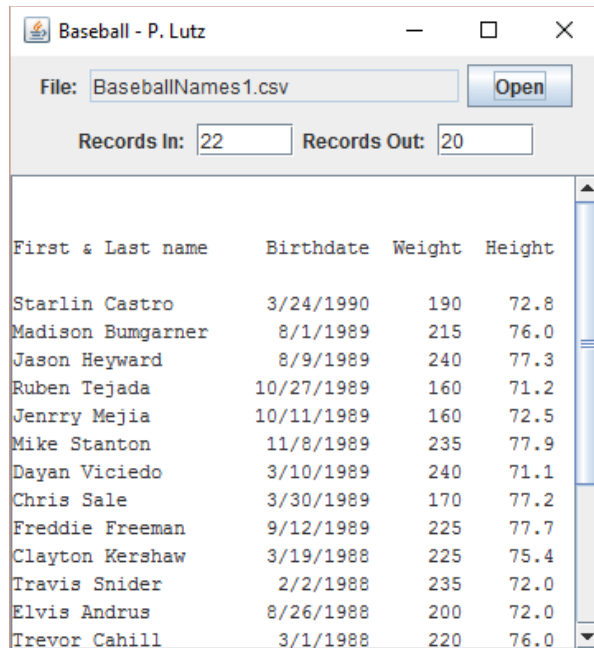
Dropbox:

Send to the dropbox your best version of the file processing. Do NOT submit more than one version, as the wrong one may be graded, lowering your grade.

This homework is due into the dropbox by the date/time shown on the dropbox.

Sample outputs:

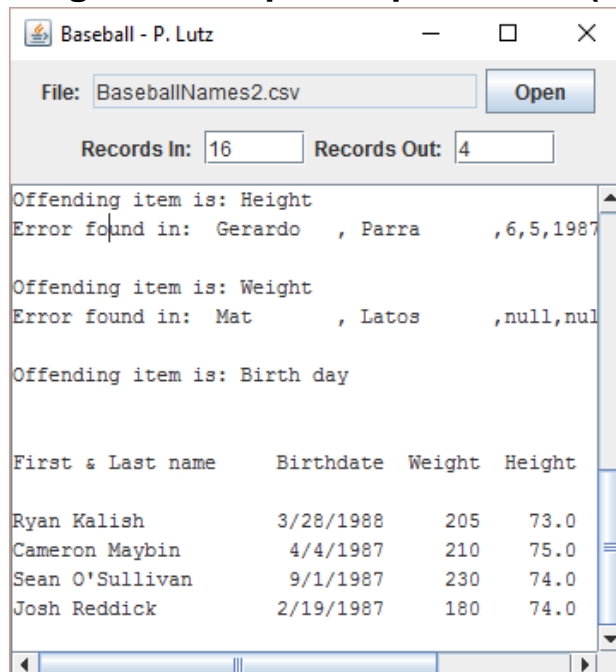
BaseballNames1.csv – no errors. Use this for initial testing only.



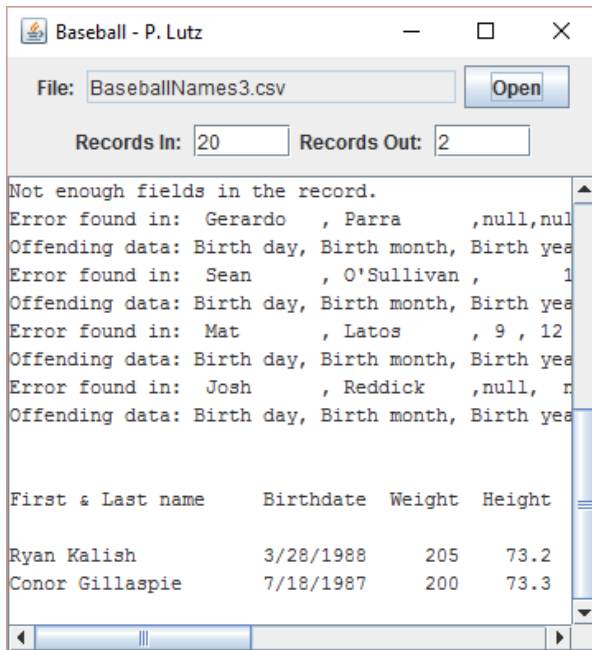
First & Last name	Birthdate	Weight	Height
Starlin Castro	3/24/1990	190	72.8
Madison Bumgarner	8/1/1989	215	76.0
Jason Heyward	8/9/1989	240	77.3
Ruben Tejada	10/27/1989	160	71.2
Jenrry Mejia	10/11/1989	160	72.5
Mike Stanton	11/8/1989	235	77.9
Dayan Viciedo	3/10/1989	240	71.1
Chris Sale	3/30/1989	170	77.2
Freddie Freeman	9/12/1989	225	77.7
Clayton Kershaw	3/19/1988	225	75.4
Travis Snider	2/2/1988	235	72.0
Elvis Andrus	8/26/1988	200	72.0
Trevor Cahill	3/1/1988	220	76.0

There are more lines than are shown here, as indicated by the vertical scrollbar.

Single error reported per record (BaseballNames2.csv)



First & Last name	Birthdate	Weight	Height
Ryan Kalish	3/28/1988	205	73.0
Cameron Maybin	4/4/1987	210	75.0
Sean O'Sullivan	9/1/1987	230	74.0
Josh Reddick	2/19/1987	180	74.0

Multiple errors reported per record (BaseballNames3.csv) and record with insufficient fields

ISTE-121 – Homework 3 – Binary I/O Name: _____

Item	Possible points	Earned points
FileChooser used to get name of csv file	5	
Parses input file data properly	10	
Handles heading and blank line as expected	5	
Correctly writes binary file (and with proper name)	10	
Reports Records In and Records Out correctly	5	
Reads binary file using appropriate DataInputStream methods	10	
Writes formatted report to screen, including dates formatted as shown Report is clean and properly aligned	15	
Complete Javadoc and comments in code	10	
All file I/O operations are properly included in the code	10	
All errors and exceptions are handled with messages via an Alert or written to taData	10	
Code works as expected	10	
Bonus points levels: Bonus 1 – Report all errors that occur on a record Bonus 2 – Report the original record once for all reported errors found in that record (includes points from bonus 1)	+5 +5	
Deductions: <ul style="list-style-type: none"> Late Coding standard violations, etc. Submitted something other than a zip file (-5) Duplication of code that has the same functionality 		
Total:	100	

Comments: