Clayton Szelestey

Software Development I

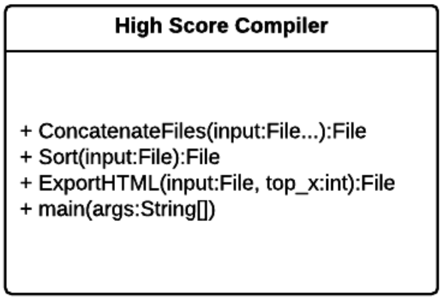
Project Milestone

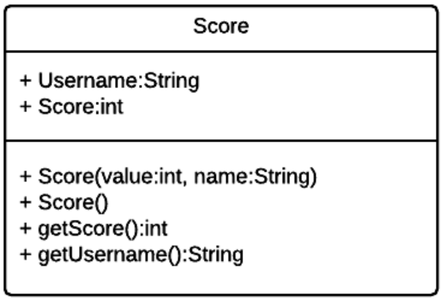
High Score Compiler - Milestone

Abstract: The program I am creating takes separate files containing high scores and concatenates them into a single high score file. It also produces an html file containing a specified number of top scores.

Introduction: The motivation for this project comes from a separate project I am working on for my Game Design II class. My game has a scoring system and players can earn high scores. I would like to use this program to take high scores from instances of my game on multiple machines and create a global high score list. The outputting in html would be, ideally, so I can easily update a website with the global high scores, and can use this program with any game that has a scoring system. Players would then be able to go to a website and check the global high scores to see where they stand.

Detailed System Description:


High Score Compiler
+ ConcatenateFiles(input:File...):File
+ Sort(input:File):File
+ ExportHTML(input:File, top_x:int):File
+ main(args:String[])



The main method will execute the program. It will ask the user for an integer to determine the number of displayed high scores in the html document. Once the number is determined the ConcatenateFiles method will execute. It will take in a number of files and join them together in a single file that is formatted the same way as the original files. Measures will be taken to ensure the program gracefully handles erroneous input of both the number of high scores to be displayed and the input files. Once a single file is created, the single global scores file will be used as input for the Sort method. The Sort method will parse through the global scores file and create an array of Score objects that will be sorted from highest to lowest score, to be used in the sorted scores file. Once the sorted file is completed, the ExportHTML method will output a separate file containing formatted html with the selected number of high scores displayed from the sorted global scores file.

Requirements:

- Take in any number of high score files within a directory (I don’t want the user to have to specify the number of files)
- Concatenate scores from all input files into a single file
- Gracefully handle problems if file format is incorrect
- Sort high scores from concatenated scores from highest to lowest using Score class

- Have concatenated score be in format that can be used again by the program
- Output a file containing the new global leaderboard
- Output a formatted html file containing a specified number of high scores to be displayed on a website

User Manual: Put all individual high score files into a specified directory. Run the program and answer the prompt asking what number of high scores you would like displayed in the html file. A file containing all global high scores will be output along with the html file containing the specified number of scores to be displayed (e.g. Top 10).

Conclusion: The goals of this project are to be able to keep track of high scores across multiple instances of my game, have a leaderboard that can be easily viewable online, and have the system work with any correctly formatted score files so that I can use the program across any future games I make that contain high scores.