SE 211: Software Specification and Design II

Software Design Document
CSVlibrary & CSVapp

Assignment #2

Cassidy Ashe

2/28/2021
Revision 0

SDD Revisions

Date	Description	Revision
2/26/2021	Created the document	0

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to describe the implementation of the CSVlibrary as described in Assignment 1, the SRS document for this project. The purpose of this application is to allow users to access and modify CSV files.

1.2 Scope

This document describes the implementation details of the CSVlibrary. CSVlibrary is a library that allows for the storage and access of CSV files with the intention of being accessed through the application CSVapp. CSVapp serves the purpose of being the user interface for the CSVlibrary application.

1.3 Definitions, Acronyms, Abbreviations

CSVlibrary The name of the product being developed to be used in conjunction with CSVapp

CSV Comma-separated Value

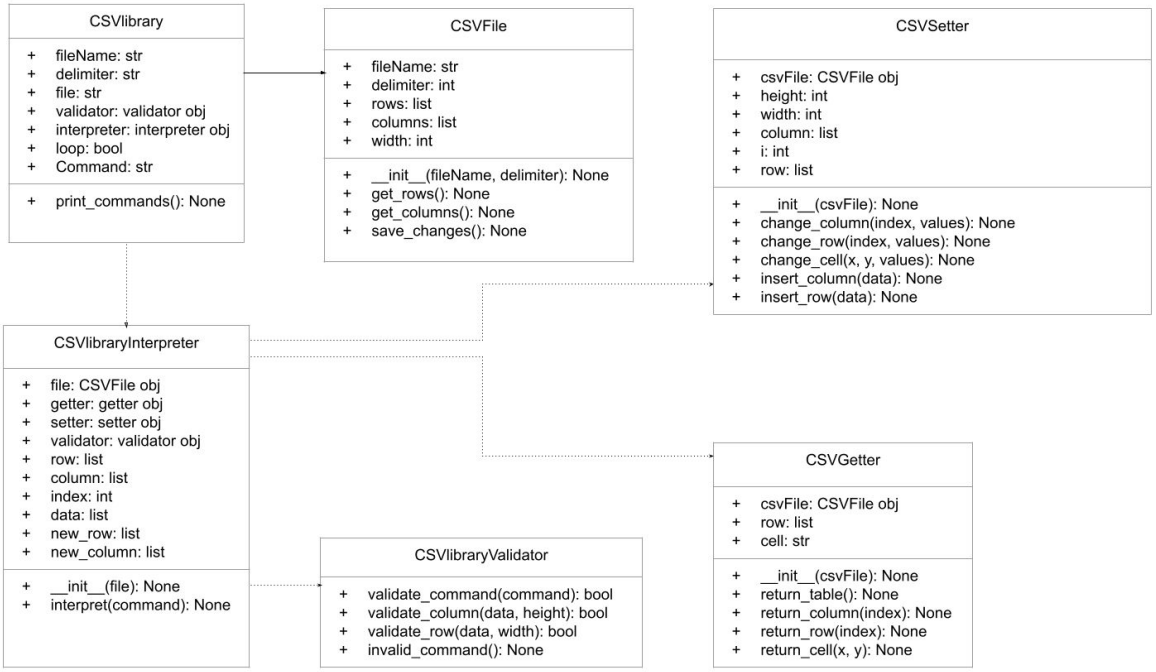
CSVapp The name of the product being developed to be used in conjunction with CSVlibrary

2 Design Overview

2.1 Description of Problem

The CSVlibrary library will serve the purpose of supporting the CSVapp, allowing users to have the ability to access their CSV files in a simple and efficient way. CSVlibrary contributes to both personal and professional uses, as it not only could be used by casual users, but has the potential to be used by businesses to effectively access and modify data.

3 CSVlibrary Interface



3.1 CSVlibrary

3.1.1 Attributes

Name	*Type*	*Description*
fileName	str	User input that determines what file to open
delimiter	str	The delimiter that will be used to separate values from the file
file	CSVFile object	An object using the filename and delimiter
validator	validator object	Instantiates validator
interpreter	interpreter object	Instantiates validator
loop	bool	Starts as True, becomes false when user inputs “Exit” command
command	str	User input that starts a command

3.1.2 Methods

None print_commands()	
Input:	None
Output:	List of commands that can be performed on the CSV file
Description:	Prints the following commands: RTAB - Return Table RROW - Return Row RCOL - Return Column RCEL - Return Cell CROW - Change Row CCOL - Change Column CCEL - Change Cell DROW - Delete Row DCOL - Delete Column DCEL - Delete Cell IROW - Insert Row ICOL - Insert Column EXIT - Exit program

3.2 CSVFile

3.2.1 Attributes

Name	*Type*	*Description*
fileName	str	User input that determines what file to open
rows	list	List of lists containing the data for each row in the file
columns	list	List of lists containing the data for each column in the file
width	int	Number of items in rows[0], describes how many columns there should be

3.2.2 Methods

None __init__(fileName:String, delimiter:String)	
Input:	The name of the file and delimiter to create a CSVFile object
Output:	A CSVFile object is created

Description:	The initializer for the CSVFile object.

None get_rows()	
Input:	None
Output:	Adds the values of each row to the rows list.
Description:	Reads rows from CSV file and adds them to rows list.

None get_columns()	
Input:	None
Output:	Adds the values of each column to the columns list.
Description:	Takes column values from rows and adds them to list.

None save_changes	
Input:	None
Output:	None
Description:	Updates changes made to data by writing to the CSV file

3.3 CSVLibraryInterpreter

3.3.1 Attributes

Name	*Type*	*Description*
file	CSVFile object	The CSV file to be used
getter	csvGetter object	Instantiates getter object to return values
setter	csvSetter object	Instantiates setter object to set values
validator	validator object	Instantiates validator object to validate commands and inputs
index	int	Index value

data	list	List of values to change/add to a row or column
row	list	A row to modify
column	list	A column to modify

3.3.2 Methods

None __init__(file: CSVFile object)	
Input:	CSVFile object to be manipulated
Output:	None
Description:	Initializes an interpreter object.

None interpret(command: String)	
Input:	A command string from user input to be interpreted.
Output:	None
Description:	Interprets and executes commands given by the user.

3.4 CSVlibraryValidator

3.4.1 Methods

Bool validate_command(command: String)	
Input:	A command string from user input to be validated.
Output:	A boolean value indicating the validity of the command.
Description:	A validation method that determines if the user input a valid command.

Bool validate_column(data: list, height: int)	
Input:	A data list from user input and height of column to be validated.

Output:	A boolean value indicating the validity of the data and height.
Description:	A validation method that determines if the user input a valid data set.

Bool validate_row(data: list, width: int)	
Input:	A data list from user input and width of column to be validated.
Output:	A boolean value indicating the validity of the data and width.
Description:	A validation method that determines if the user input a valid data set.

None invalid_command()	
Input:	None
Output:	A print to the user indicating the input of an invalid command.
Description:	A method that serves the purpose of being an error message.

3.5 CSVGetter

3.5.1 Attributes

Name	*Type*	*Description*
csvFile	CSVFile object	The CSV file to be used
row	list	A row to return
cell	str	A column to return

3.5.2 Methods

None __init__(csvFile: CSVFile object)	
Input:	CSVFile object to retrieve data from.
Output:	A CSVFile object.
Description:	Initializes a CSVFile object.

None return_table()	
Input:	None
Output:	Returns the CSV file data to the user.
Description:	A method to return the table created in the CSV file.

None return_column(index: int)	
Input:	Index of the column.
Output:	Returns the contents of the row to the user.
Description:	A method to return a specific column from the CSV file.

None return_row(index: int)	
Input:	Index of the row.
Output:	Returns the contents of the column to the user.
Description:	A method to return a specific row from the CSV file

None return_cell(x: int, y: int)	
Input:	Row (x) and column (y) of the cell that is meant to be modified.
Output:	Returns the contents of the cell to the user.
Description:	A method to return a specific cell from the CSV file

3.6 CSVSetter

3.6.1 Attributes

Name	*Type*	*Description*

csvFile	CSVFile object	The CSV file to be used
height	int	Length of first column, determines number of rows
width	int	Length of first row, determines number of columns
column	list	A list of data to be modified
i	int	Index value
row	list	A list of data to be modified

3.6.2 Methods

None __init__(csvFile: CSVFile object)	
Input:	A CSVFile object to be manipulated.
Output:	A setter object to make changes to the CSV file.
Description:	Initializes the setter object.

None change_column(index: int, values: list)	
Input:	The index of the column that the user would like to change as well as the values to change that column to.
Output:	None
Description:	Updates the CSV file with whatever valid modifications the user has made.

None change_row(index: int, values: list)	
Input:	The index of the row that the user would like to change as well as the values to change that row to.
Output:	None
Description:	Updates the CSV file with whatever valid modifications the user has made.

None change_cell(x: int, y: int, values: list)	
Input:	The cell (row, column) that the user would like to change as well as the value

	to change that cell to.
Output:	None
Description:	Updates the CSV file with whatever valid modifications the user has made.

None insert_column(data: list)	
Input:	A list of data in order to create a new column.
Output:	None
Description:	Updates the CSV file with whatever valid modifications the user has made.

None insert_row(data: list)	
Input:	A list of data in order to create a new row.
Output:	None
Description:	Updates the CSV file with whatever valid modifications the user has made.

6 References

[Sample Design Document, BBLearn]