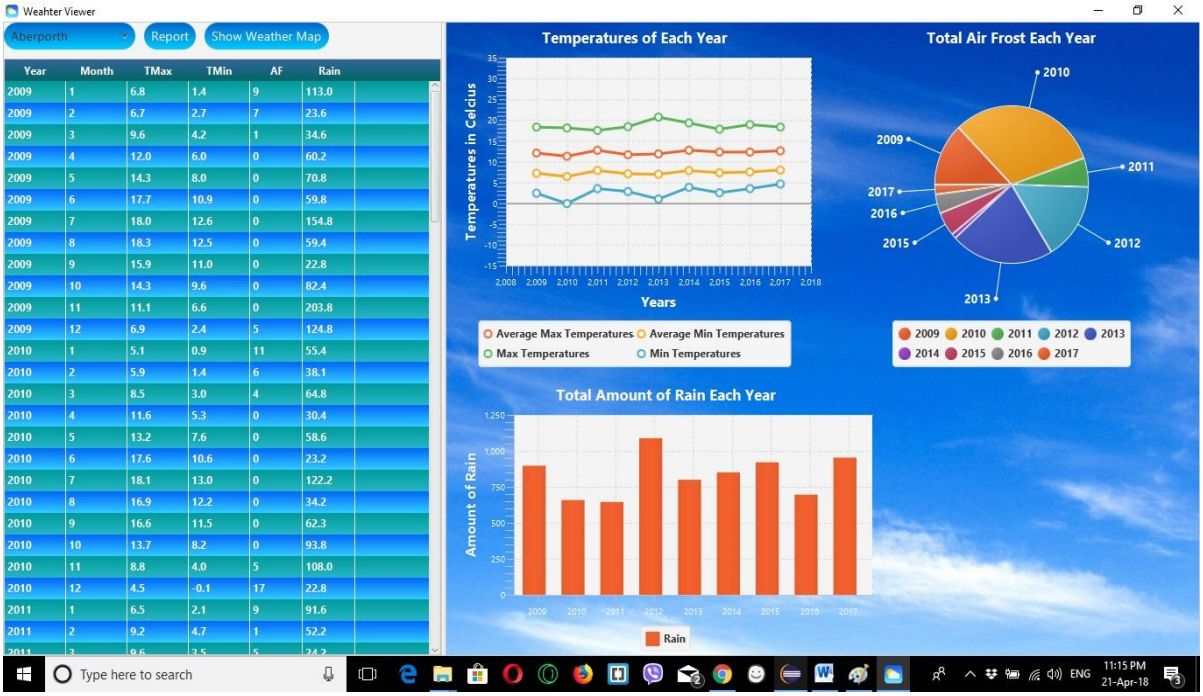
Weather Viewer App Report

Introduction

The application reads csv files separately to a table view and displays the data to different charts. The users can see a report in a text file after the press the report button. Next to the report button there is another button that opens a new window and the users can see live weather conditions from around the world. The users can select from the 1st button any weather station to see the data. When it is clicked the table view changes along with the charts. The window is resizable and there is also a background image of a sky at the charts area.

Main Page



In the main class I initialized the TableView and the Observable list. Then inside the start method I created 6 table columns, set the items for tableview and added the columns to tableview. Following I created 3 charts: Line Chart for the temperatures because I display 4 different data, Pie Chart for the Air Frost so we can see which year had the biggest period of air frost and a bar chat for the rain to see which year had the most amount of rainfall.

Then below the charts I created the buttons. I have a choicebox in order to add all the stations names and be able to click whichever station the users want. I have set the initial value to "Aberporth" so when the app opens the button shows that name. After that is the report button which when the users click it opens a text file and displays the data from the current station which is displayed on the screen. Then is the last button of the app which when it is clicked a new window opens with a global map and shows the current weather conditions of cities around the world. I created a HTML file and I used 2 API's: Google maps and https://openweathermap.org/. And then with a method that I created I called added this HTML to my app. I added all the buttons

inside an HBox to align them horizontally. I set some spacing and created a css class so I can use it to style the buttons.

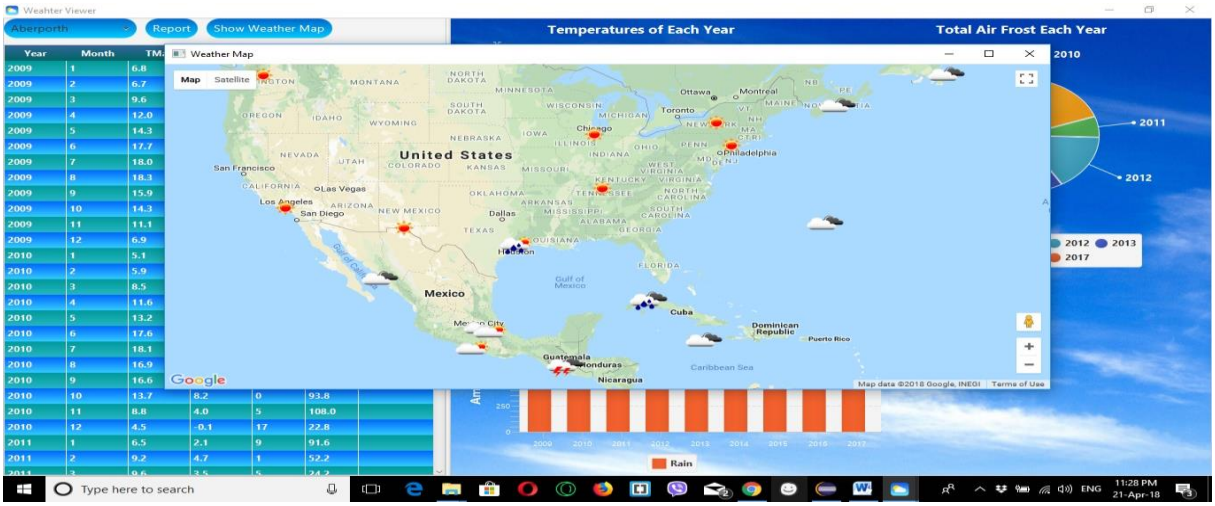
For the tableview I created a VBox to put it under the buttons. Then I created another VBox to put all the charts. Inside that VBox I created an HBox and added 2 charts the linechart and the piechart so it can be aligned horizontally. The 3rd chart I put inside another HBox and it was placed under the other 2 charts. Finally I added them to the main VBox and gave them some spacing.

Below these I adjusted the width of the columns and the height of the table view so it can take up the whole height of the window. Then I used the SplitPane to divide the screen in the middle so the users can see both the table view and the charts or they can increase the size of whichever window they want. I gave more space to the charts so the users can see them clearly. In the SplitPane I added all my elements. Then I created the Scene and the Stage and added all my elements. I also add an icon for the task bar using the stage.getIcons(). Below all these there is the section where I call all the methods that I have created inside the buttons. I also call the methods outside of the buttons on order for the program to be able to run when it opens for the first time.

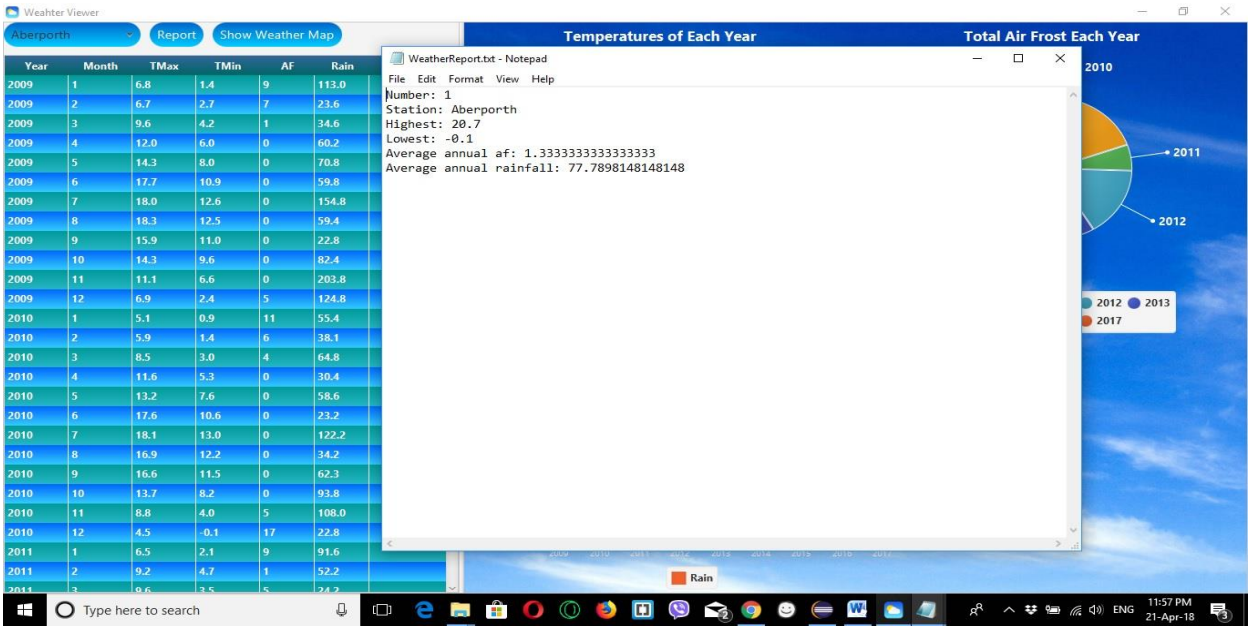
Next is the Record Class in which I set the constructor with the values for the table view and I set the getters for thisvalues.

Next is the Charts class one of the most important class of the program. Here there are the methods that make the charts work. First is the printChart method that along with the readTableData fill up the line chart with the average max and min temperature and the highest and lowest temperature each year. In the readTableData I loop through the tableView. Then I created a sub list and depending on the range of the months or years that we want to find for example the average Tmax we give a starting and ending point. In our example the range is 12 months (1 Year). I did the same the same principles apply to readTableDataMaxTemp returns the maxTemp of the year, readTableDataMinTemp returns the min temp of the year and readTableDataRain returns the total amount of rain year (the same method I used to print the air frost for the pie chart). Then the printChart method loops 9 times (9 Years data) and I print the result for each year. The same applies to the rest of the print charts method with some changes.

Next is the WeahterWindow method which creates a new window with a live weather conditions map. I used the javafx components WebView , WebEngine and the File to load the HTML.



The next class is the WriteFIle. This class has the writeFile method which takes the data which are being displayed that moment and writes them to a file and opens that file. To do that first it loops through an array with the stations name and if it is equal to the name in the choicbox then it writes the index which that station is to the file I added 1 more string inside that array so that the 1st station index starts from index [1]. Then method It loops through the tableVIew data and gives back the max and min temperature and the average of air frost and rain of all years. After that I create a file and inside that file I write everything and when it is done the file opens and the users can see the data.



Year	Month	TMax	TMin	AF	Rain
2009	1	6.8	1.4	9	113.0
2009	2	6.7	2.7	7	23.6
2009	3	9.6	4.2	1	34.6
2009	4	12.0	6.0	0	60.2
2009	5	14.3	8.0	0	70.8
2009	6	17.7	10.9	0	59.8
2009	7	18.0	12.6	0	154.8
2009	8	18.3	12.5	0	59.4
2009	9	15.9	11.0	0	22.8
2009	10	14.3	9.6	0	82.4
2009	11	11.1	6.6	0	203.8
2009	12	6.9	2.4	5	124.8
2010	1	5.1	0.9	11	55.4
2010	2	5.9	1.4	6	38.1
2010	3	8.5	3.0	4	64.8
2010	4	11.6	5.3	0	30.4
2010	5	13.2	7.6	0	58.6
2010	6	17.6	10.6	0	23.2
2010	7	18.1	13.0	0	122.2
2010	8	16.9	12.2	0	34.2
2010	9	16.6	11.5	0	62.3
2010	10	13.7	8.2	0	93.8
2010	11	8.8	4.0	5	108.0
2010	12	4.5	-0.1	17	22.8
2011	1	6.5	2.1	9	91.6
2011	2	9.2	4.7	1	52.2

The last class is the ReadCSV which has the readCSV method another important method which reads all the csv files and displays the data on the tableView. There is a for loop that clears all the data from the table vie every time the user wants to load a new station by clicking the choicebox. Then a string variable is set with the path of the files and I concatenated with the parameter name which is a string which takes the current value from the choicebox. And then lastly with the buffer reader and the delimiter which I created I read and split each word.

The last file is the table.css which I used to style the entire app. I used gradient colors to change the colors of the table. I changed the color of the arrow and fonts to white. I loaded an image as a background for the charts using the #vbox-custom id. And I also styled the charts's axis, labels and the buttons. I gave the buttons rounded shape, changed the color and added a hover effect.