Code Documentation

Main.java

Contains the 'main' method. It issues calls to various methods to perform the following functions:
- Read data from input file
- Create a visualisation of list of states
- Create a tree-map of the data
- Create a mashup for the data

Readfile.java

This class reads the input data file, in Excel format, into a Prefuse Table.

Control enters this class from the 'run' method, which opens the input Excel file, creates a new table 'table', and issues calls to enter data into the table from the worksheet.

Method 'getCellContents'
Signature: String getCellContents(int row , int col)
Takes a row number and column number, and returns the data at the specified position in the input file as a String.

Method 'getCellType'
Signature: Class getCellType(int row , int col)
Takes a row number and column number, and returns the type of the data element at the specified position in the input file, as a Java Class.

Method 'addColumns'
Signature: void addColumns()
Takes no input. It determines the number of columns in the input file, and adds a column to 'table' corresponding to each column in the input file.
Returns no output.

Method 'addRow'
Signature: void addRow(int row)
Takes a row number, and adds the data from the cells in that row from the input file, into a new row in 'table', after some preprocessing (like removing the "%" sign in the attendance column).

Method 'addRows'
Signature: void addRows()
Takes no input. It determines the number of rows in the input file. For each row, a new empty row is added to the table 'table', and the method 'addRow' is invoked to populate the cells of the row.

StateMenu.java

This class creates a Fisheye Menu visualisation of all the states. On clicking a state, a Fisheye menu of the political parties in the state is displayed. The list of MPs of a given political party from a state can be seen by clicking the name of the political party in the Fisheye menu.

Control enters this class from the 'run' method, which makes necessary function calls to create the visualisation.

Method 'addMenuItem'
Signature: void addMenuItem(String name, ActionListener listener)
Adds a row to the list of items and binds to it the action and the label.

Method 'firstScreen'
Signature: StateMenu firstScreen(TreeSet<String> tree, final Table tab,final int row)
Takes the list of states as a TreeSet, and displays the states as a fisheye menu. It returns the visualisation as a StateMenu.

Method 'secondScreen'
Signature: StateMenu secondScreen(TreeSet<String> tree, final Table tab,final int row, final String stateCl)
Takes the list of political parties as a TreeSet passed to it by the method firstScreen when a State name is clicked in the first screen, and displays the parties as a fisheye menu. It returns the visualisation as a StateMenu.

Method 'thirdScreen'
Signature: StateMenu thirdScreen(TreeSet<String> tree, final Table tab,final int row, final String partyCl, final String stateCl)
Takes the list of members of parliament of a given party from a given state as a TreeSet passed to it by the method secondScreen when a Political Party name is clicked in the second screen, and displays the MPs as a fisheye menu. It returns the visualisation as a StateMenu.

StateMenu class contains a nested class VerticalLineLayout, which lines up all VisualItems vertically. It scales the size of the layout such that all items fit within the maximum layout size, and updates the display to the final computed size.

TreeMap.java

This class is responsible for creating a TreeMap visualisation of the given dataset. The Tree which it visualises is structured as follows:
- The first level contains the name of the states
- Each state has as its children the political parties in that state, as the second level of the tree
- Each political party has as its children the MPs in of that political party from the state, as the third level of the tree

The nodes corresponding to MPs are color coded according to the political parties to which they belong, with the same color for a given party across all states.

Method 'putActions'
Signature: void putActions()
This method binds the actions to the visualisation.

Method 'makeData'
Signature: Tree makeData()
This method parses the table created after reading the input file, and constructs a tree with states at first level, political parties of a given state at the second level, and MPs of a given party from a given state at the third level. It also calls the method colorCoding which assigns a unique color to each political party.
Returns the constructed tree.

Method 'display'
Signature: JComponent display(Tree tm, Hashtable<String,int[]> colorCoding)
This method takes a tree and a color coding, and returns a JComponent constructed from them.

Method 'createColorCoding'
Signature: Hashtable<String,int[]> createColorCoding(TreeSet<String> t)
Takes a list of String values as a TreeSet, and creates a hashtable which maps each String to an array of three integers containing RGB values of unique color assigned to it.

MakeSummaryTable.java

This class reads the table created after reading the input file, and constructs a new table that contains the aggregate values of various attributes grouped by states and political parties.

Control enters this class from the 'make' method.

Method 'putColumns'
Signature: void putColumns()
Adds columns to the summary table created.

Method 'addRowToTable'
Signature: void addRowToTable(String state, String party, Integer attendance, Integer questions, Integer age)
Takes the values of various attributes as read from 'table', and updates the row of the summary table with the same StateName and PartyName, by adding the passed values to the current values in the table.

Method 'addRows'
Signature: void addRows()
Reads each row of 'table', and invokes the function addRowToTable().

Method 'addRow'
Signature: void addRow(String state, String party, int stateIndex)

Takes the values of StateName, PartyName and StateIndex. Adds a row to the summary table with the values of other aggregate attributes initialised to zero.

Method 'removeNullRows'
Signature: Table removeNullRows()
Creates a new table from the summary table, after filtering out the rows for which the aggregate attributes have a value of zero.

Mashup.java

This class creates a mashup visualisation of the given data, using the summary table.
The attributes to be plotted on X and Y-axes, and the attribute according to which the points are color-coded can be chosen at runtime.

Method 'display'
Signature: void display(Table t)
Creates a JFrame to display the scatter plot.

MP.java

This class declares a datatype representing a member of parliament, containing all the details of an MP.