Group 1

Aaron Hirvi

Miikka Venäläinen

Onni Merilä

Samundra Dhakal

Design document

Components:

- MenuWindow
    - A view for selections
    - In this window user can select the data to be shown

Classes:

- Main
    - Starts the program
    - Opens menu view
- MenuWindowController
    - Creates a menu view
    - Calls the appropriate class based on the task selection
- RoadDataScene
    - Needs location and time period information – gets it from the menu view. For the forecast there is a seperate time selection
    - Asks for specification of what kind of data user wants to see and what are the search parameters. These are asked via dropdown menu selections where user can select one option (maintenance or condition forecast) and several options (visibility, friction, precipitation, winter slipperiness, overall road condition)
    - Creates a scene based on the data and parameters and injects it into menu view
    - User can make alterations to the data shown
    - When the graph is drawn, save button appears
- WeatherDataScene
    - Needs location and time period information – gets it from the menu view
    - Asks for specification of what kind of data user wants to see and what are the search parameters. These are asked via dropdown menu selections where

user can select several options (temperature ,observed wind, observed cloudiness, predicted wind, predicted temperature)
    - Creates a scene based on the data and parameters and injects it into menu view
    - User can make alterations to the data
    - When the graph is drawn, save button appears
- CombinedDataScene
    - Needs location and time period information – gets it from the menu view
    - Gets weather information from WeatherDataScene and road data from RoadDataScene and combines it
    - Data can be altered after it has been created
    - Save button appears when graph is drawn
- APICall
    - Provides data from digitraffic and FMI to other classes
    - Has a getRequest() function which handles the API calls and returns a JSONObject. Function uses org.json to handle JSON data and other function to convert XML to JSON

Dependencies:

- org.json
    - We decided to use org.json as our JSON and XML parser class, as it had both JSON and XML support.
- JFreeChart
    - Used for drawing beautiful plots used in the program. We decided to use it based on the look of the example images, and the fact that it has existed for a long time and still getting regular updates.
- APIs
    - https://www.digitraffic.fi/en/road-traffic/
    - https://tie.digitraffic.fi/swagger/
    - https://en.ilmatieteenlaitos.fi/open-data-manual
    - Program needs these APIs to get all the needed and wanted data.

Traffic:

- User click on the traffic menu, they will be redirected towards the selection view of the traffic.
- User can select the traffic using either by selecting the point or via selecting the road.

Case One: Selection Of the Road

- When user click on the select a road, the window appear with the input field for providing or selecting the road number.
- Once road number is provided, the data will be fetched using the API in the window which included the message of traffic updates.
- These updates will be displayed in the right side of the screen.
- If the user wants to have others information, there is the drop-down selection in the left side of the screen which includes:
    - Temperature, Observed wind, Observed cloudiness, Predicted wind, Predicted Temperature and so on.
- Once user selected the data from the drop-down, the info will be visible in the graph plot.

Case Two: Selection Of Point

- User choose the select the point, they will view the UI with input fields for providing the coordinates of road as well as the search button to get the info from the server.
- Once user click on the search button, the info message should be visible in the right side of the UI.
- In the left side of the UI, there's present the drop-down selection from where user can request for the additional information just like in the case one and it should be visible in the UI.