

Design document – Mid-term submission

Henriikka Rikala, Matias Nurmi, Eetu Häkkinen ja Mikko Haavisto

Contents

| | |
|------------------------------|---|
| Decision making process..... | 2 |
| Idea for our software | 2 |
| Classes..... | 3 |
| Self-evaluation | 4 |
| Sources..... | 4 |

Decision making process

Our group has decided to use C++ and Qt to do our software. Prototyping was done using Figma. Those were chosen mainly because we were familiar with them.

As we used Qt our design follows MVC-pattern. Some Qt elements already use MVC, and it felt natural to design our software to follow it.

For displaying the data as graphs, we used QtCharts-module. It was presented in this course and seemed like the best approach.

Qt doesn't provide the selection type we wanted so we decided to use opensource code from the internet to create multiple selection combobox. It was altered to fit our needs. [1]

Idea for our software

This part is a short description of how our design works. The idea is also visualized in the prototype for the parts that are still not implemented.

- First user sees the starting page which shows no data yet. The header of the page shows the selections that the user needs to do to show data.
 - There are boxes for selecting data, measurement station and the time period
 - From plus-sign user can submit choices
 - From :-sign user can take closer look on settings
- When user selects data and station on certain time and presses +-sign, data is shown. If more than one station is selected, those will be shown on the same chart. If more data is selected it is shown in its own chart. (There is also a possibility for multiple stations)
- From settings (: -button) user can see the current selections, there is also options for looking history and saved selections.
- In current selections
 - User can remove selections from red trash cans
 - User can save current selections from save selections
 - User can clear all selections from clear selections
- In history selection history is shown
- In bookmarks

- User can check saved selections
- User can delete saved selections
- User can upload selections to show the charts

Classes

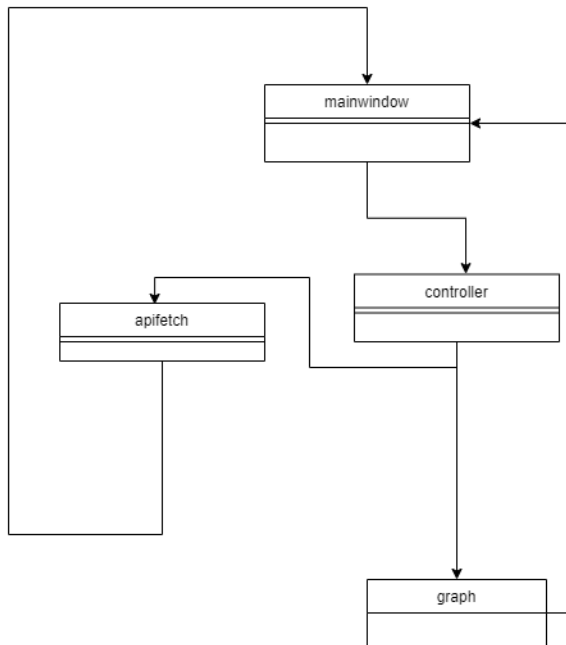


Figure 1 Class diagram

Figure 1 presents simple class diagram of our software.

MainWindow

Mainwindow is the view of our design. It communicates with controller, and it shows wanted UI.

ApiFetch

ApiFetch communicates with controller. It does the wanted API calls to get right data and then handles the data. It returns the data to mainwindow.

Controller

Controller communicates with mainwindow, ApiFetch and Graph. It uses ApiFetch as data handler and Graph to create graphs.

Graph

Creates graphs when the controller asks. Sends data to mainwindow

Self-evaluation

How our design has supported implementation

The implementation phase has been easier because everyone has been on the same page about how things are going to work. We also think that prototyping has helped us to implement the software since we had a concrete model of what we were trying to do.

How well have we been able to stick to our original design

We have been able to stick to our design very well. We thought the ideas very thoroughly while doing the prototype, so we didn't have to do it anymore in the implementation phase. We also discussed how things are possible to do while designing the prototype, so we were able to follow the prototype.

How well have we been able to implement features based on our original plan

Since we already discussed possible implementation ways for some of the main parts in prototyping phase our prototype was easy to implement. We did run into some changes during implementation so not everything is exactly like we thought before.

For example, we prototyped the selections menu in a way that selections could be changed in the settings section. When implementing it we thought that it would make the software unnecessarily complicated and doing it would have been harder. We thought it would be enough to show selections and to be able to remove them one by one.

We also added pages to the implementation. Now our graphs can be shown on multiple pages instead of all showing at once.

How does our design correspond to quality

We have tried to make UI easy to use and clear to read. We thought of those things already when prototyping and in implementation we have evaluated how well the prototyped features function. One important part we focused on was to prevent user from doing things that would cause errors.

What changes are we anticipating we need to make to be able to implement the remaining features

We still need to implement some features. History, bookmarks and downloading a graph are a few features we are planning to do. Also, some UI improvements like axels on graphs and some buttons. Those changes are probably possible to implement as prototyped. Some changes might occur if something won't work as well as we thought.

Sources

[1] <https://github.com/ThisIsClark/Qt-MultiSelectComboBox>