

Map of Denmark

*First-Year Project, Bachelor in Software Development,
IT Univ. of Copenhagen*

Group 12

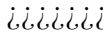
Jakob Melnyk jmel@itu.dk

Niklas Hansen nikl@itu.dk

Emil Juul Jacobsen ejuu@itu.dk

Jens Dahl Møllerhøj jdmo@itu.dk

Supervisors: Lars Birkedal

Advisors: Jonas Brabrand Jensen and Filip Sieczkowski  64a14081a839b949f9c39c

May 25th, 2011

Contents

1	Preface	2
2	Background	3
2.1	Problem area	3
2.2	Our requirements	4
2.2.1	Project requirements	4
2.2.2	Our own requirements	4
2.3	Data set	4
2.3.1	Graph	4
2.3.2	UTM-Coordinates	4
2.4	MVC structure	4
3	UI-analysis	5
3.1	Interesting features	5
3.1.1	Pins	5
3.1.2	Navigation	5
3.1.3	Bike/car	5
3.1.4	Hotkeys	5
3.2	Features we did not implement	5
3.2.1	Choice of roads to be displayed	5
3.2.2	Smooth scrolling	5
3.2.3	Dynamic route finding	5
4	Implementation	6
4.1	Dijkstra vs A-star	6
4.2	Evaluator	6
4.3	Quadtree	6
4.4	Serialization	6
4.5	UTM-conversion	6
4.6	Mousezoom	6
4.7	Floats	6

5	UML-diagrams	7
5.1	MVC	7
5.2	Simple Diagram	7
5.3	Control flow	7
6	Tests	8
6.1	WB: closestEdge	8
6.2	JUnit	8
6.3	System test	8
7	Manual	9
7.1	Navigation	9
7.1.1	GUI	9
7.1.2	Keyboard	9
7.2	Zoom	9
7.2.1	GUI	9
7.2.2	Keyboard	9
7.3	Route find	9
7.4	Bike/car	9
7.5	Resize	9
7.6	Road display	9
8	Product conclusion	10
9	Group norms	11
10	Diary	12
11	Worksheets	13
12	Process description and reflection	14

Chapter 1

Preface

1

Chapter 2

Background

2.1 Problem area

Over the last decade people have switched from traditional roadmaps to using the web-maps. This is a change without any negative side-effects. The online services remove all the problems with determining the quickest route between two points and you spend no time browsing the pages of the map to find what you need. With the popular smartphones the online map is even more useful, since you no longer need to prepare your trip before you leave. The online maps have now been used for many years and haven't been slow at adopting new features to improve their usability. They have both implemented satellite-maps that allow us to browse the entire planet from above, and lately the feature called Google Street View has upped the stakes when allowing us to look at any direction from a given point of a road. The two maps that we use the most are Google Maps and the Danish map called Krak. These maps both have the mentioned features but slight differences in the way the user navigates and searches for routes. Because of the widespread knowledge of the online maps, the users have been accustomed to certain features and ways of using the map. It is very important that we, with a new map program, use this knowledge to our advantage and don't try to reinvent the wheel. By using some of the commonly used controls in our map, a user will be able to quickly adapt to our program and use it efficiently.

2.2 Our requirements

2.2.1 Project requirements

2.2.2 Our own requirements

2.3 Data set

We have been provided with a dataset of roads and intersections in Denmark from Krak. Additionally we got some code for loading the data in from the text files. We have only made minor changes to the code for loading the data.

2.3.1 Graph

When the data has been loaded it is stored as a Graph containing KrakNodes and KrakEdges. The KrakEdges are the road segments and contains the name of the road, an estimated drive time, a direction of traffic and references to the two KrakNodes that are at either end of the road. The KrakNode itself contains only the coordinates for the point. The Graph itself contains a number of useful methods for searching the data like getting all edges that is connected to a KrakNode. We will be using these methods extensively throughout the project both for drawing the map and for finding the route between two points.

2.3.2 UTM-Coordinates

It is important to note that the KrakNodes are in UTM-32 coordinates. When using the UTM standard the origo is placed at the south-west corner. These coordinates need some conversion when using in Java since the origo is placed differently.

2.4 MVC structure

Chapter 3

UI-analysis

3.1 Interesting features

3.1.1 Pins

3.1.2 Navigation

3.1.3 Bike/car

3.1.4 Hotkeys

3.2 Features we did not implement

3.2.1 Choice of roads to be displayed

3.2.2 Smooth scrolling

3.2.3 Dynamic route finding

Chapter 4

Implementation

4.1 Dijkstra vs A-star

4.2 Evaluator

4.3 Quadtree

4.4 Serialization

4.5 UTM-conversion

4.6 Mousezoom

4.7 Floats

Chapter 5

UML-diagrams

5.1 MVC

5.2 Simple Diagram

5.3 Control flow

Chapter 6

Tests

6.1 WB: `closestEdge`

6.2 JUnit

6.3 System test

Chapter 7

Manual

7.1 Navigation

7.1.1 GUI

7.1.2 Keyboard

7.2 Zoom

7.2.1 GUI

7.2.2 Keyboard

7.3 Route find

7.4 Bike/car

7.5 Resize

7.6 Road display

Chapter 8

Product conclusion

1

Chapter 9

Group norms

Chapter 10

Diary

3-4

Chapter 11

Worksheets

4-5

Chapter 12

Process description and reflection

1