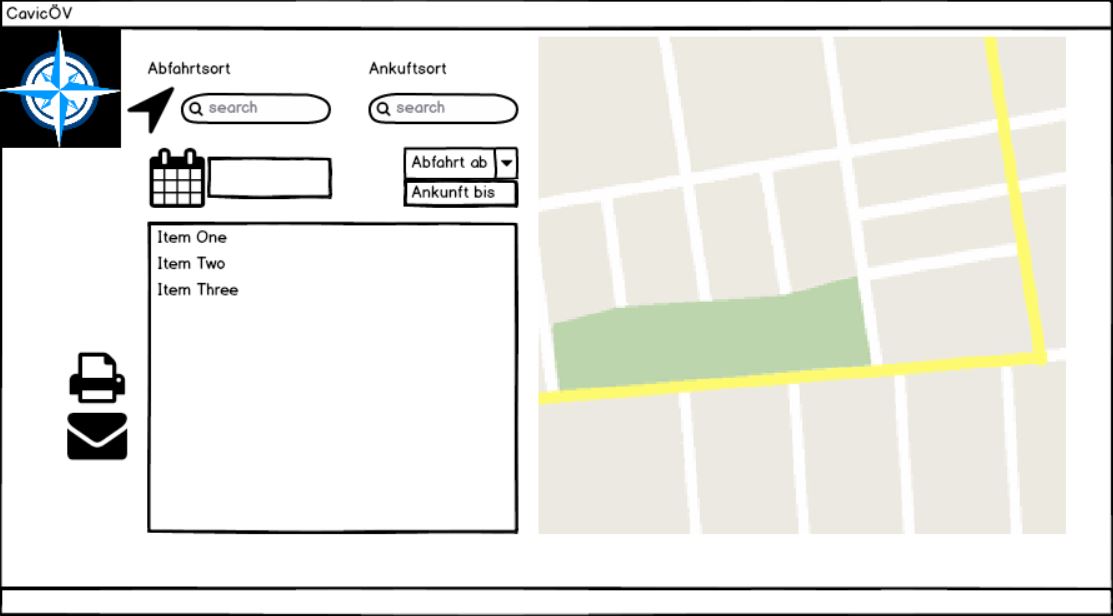
CavicÖV



Projektarbeit ÜK M318

Autor: Daniel Cavic

Version: 1.1

Datum: 14.11.2017

Inhaltverzeichnis

[Einleitung 3](#_Toc499033790)

[Ziel 3](#_Toc499033791)

[Code Richtlinien 3](#_Toc499033792)

[Umgesetzt 4](#_Toc499033793)

[Anforderungen 4](#_Toc499033794)

[1 & 4 Anforderung 4](#_Toc499033795)

[2 Anforderung 4](#_Toc499033796)

[3 Anforderung 4](#_Toc499033797)

[5 Anforderung 4](#_Toc499033798)

[6 Anforderung 5](#_Toc499033799)

[7 Anforderung 5](#_Toc499033800)

[8 Anforderung 5](#_Toc499033801)

# Einleitung

Ich habe den Auftrag bekommen, dass ich eine Applikation entwickle, welche die Verbindungen innerhalb der Schweiz mit dem öffentlichen Verkehr. Das Projekt wäre optimal, wenn man es schaffen würde, dass man die Verbindung per Mail verschicken kann. Den aktuellen Standort sollte man auch einfügen können. Es muss auch geachtet werden, dass man eine gute Codequalität hat. Abstürze sollen vermieden werden und das GUI soll Benutzerfreundlich sein.

# Ziel

Mein Ziel ist es das ich mindestens alle Forderungen mit der Priorität 1 und 2 umsetzen kann und eine Forderung mit der Priorität 3. Ich möchte mich nach meinen Coderichtlinien halten und einen Qualität reichen Code haben.

# Code Richtlinien

Lokale und globale Variablen werden klein geschrieben und gleich auf einen bestimmten Wert gesetzt. Die Variablen werden am Anfang vom Programm deklariert. 🡪 Beispiel int zahl = 0;

Bei Klassen wird der Anfangsbuchstabe grossgeschrieben bzw. nach PascalCase Richtlinien . 🡪 Beispiel: public class Person oder public class PersonInfo

Membervariablen innerhalb einer Klasse werden mit einem m\_ deklariert. 🡪 Beispiel: private int m\_ausgerechneteZahl

Eigenschaften und Methoden werden alle nach den camelCase Richtlinien geschrieben. 🡪 Beispiel: privat int ausgerechneteZahl oder public vo id zahlAusrechnen

Gui-Controls bekommen am Anfang immer einen Typ in Abkürzung und am Schluss folgt ein sinnvoller Name. Die Schreibweise entspricht den camelCase Richtlinien. Button=btn, Textbox = txt, Label = lbl, Panel = pal, usw. 🡪 Beispiel: txtName (Typ: Textbox)

Kommentare werden oberhalb des Codes geschrieben, dass man so den Kommentar auf einem Bildschirm betrachten kann, welcher auch etwas kleiner ist. Man kommentiert, was die Methode macht. Die Methoden werden kommentiert und schwere Codeausschnitte. Kommentare macht man immer mit /// <summary> über öffentlichen Methoden. Sonstige Methoden kommentiert man mit /\* am Anfang und am Schluss vom Kommentar mit \*/ aus. 🡪 Beispiel /\* Zahl wird ausgerechnet\*/

Man verwendet if-else und keine switchs und cases. Für if-else, for, foreach, usw. verwendet man den folgenden Aufbau :

if (condition)

{

Console.WriteLine("The variable is set to true.");

}

else

{

Console.WriteLine("The variable is set to false.");

}

## Umgesetzt

Ich habe meine Ziele erreicht und zum Teil überschritten. Die ersten drei Anforderungen waren nicht so einfach, aber machbar. Danach wurde es für mich recht schwer. Ich musste viel im Internet suchen und zu Hause auch arbeiten. Ich habe alle Anforderung erreicht, bis auf die, wo man die Verbindungen als Mail verschicken kann.

# Anforderungen

## 1 & 4 Anforderung

Diese Anforderungen habe ich mit Hilfe von der Methode GetStation gemacht, welche von der Klasse Transport schon vordefiniert ist. Am Anfang musste ich die Suche mit einem Button bestätigten und das Resultat wurde in eine Textbox geschrieben. Danach habe ich die vierte Anforderung implementiert und konnte die Buttons löschen. Ich habe ein User Control erstellt mit einer Textbox und einer Listbox. Der Name von der User Control Klasse ist autoSuche. Die Listbox habe ich unter der Textbox versteck. Durch dies sieht mein User Control so aus, als wäre es eine Combobox. Die listbox erscheint unterhalb der Textbox mit ihren Resultaten. Danach kann man mit Hilfe der Pfeiltasten in die List und mit Enter oder per Maus wählt man eine Station aus.  
Wenn man etwas nicht auswählt mit der Maus oder der Enter-Taste, so wird die ListBox nicht zugeklappt, bis man etwas auswählt.

## 2 Anforderung

Die zweite Anforderung habe ich mit einem Button und den zwei User Controls gelöst. Der Button wird erst eingeblendet, wenn bei User Controls etwas haben in der Textbox. Die Verbindungen werden danach in einem DataGridView eingeblendet. In der DataGridView wird die Dauer, Abfahrtszeit, Ankunftszeit und die Verspätung von der Verbindung eingeblendet. Ich habe die Methode GetConnections in der Klasse Transport verwendet. Ich habe noch zusätzlich eine Klasse Meine Verbindungen erstellt. In Der Klasse werden die Daten gespeichert, welche danach ausgegeben werden. Ich erstelle Objekte der Klassen und eine Liste. Die Liste wird danach eben ausgegeben im DataGridView.

## 3 Anforderung

In der Klasse Transport ist eine Methode erhalten, welche GetStationBoard heisst. Diese Methode gib eine Abfahrtstafel zurück. Die Abfahrtstafel gebe ich ihm DataGridView aus. Folgende Sachen werden ausgegeben: Nummer vom Fahrzeug, Zielstation, Anbieter und die Kategorie. Die ID von der Station lese ich aus, indem ich GetStation verwende und danach aus der herausgelesenen Station die ID herausgefunden mit Hilfe von einer For-Schleife.

## 5 Anforderung

Ich habe die URL von der API erweitert. Ich entnehme aus dem DateTimePicker von meiner MainForm die Zeit und das Datum. Danach übergebe ich das Datum und die Zeit der Methode GetConnections. Dort habe ich noch &datetime + Datum und Zeit ergänzt. Somit kann man seine zukünftigen Abfahrten planen.

## 6 Anforderung

Ich habe einen Button erstellt, mit welchem man herausfinden kann, wo die Station ist. Die meisten Stationen haben ein Komma zwischen Stadt und Ort. Ich teil den String beim Komma mit Split. Danach gebe ich es an die Google Maps URL weiter und öffne es auf dem Standardbrowser vom PC. Ich wollte es zuerst direkt in meiner MainForm öffnen doch dies ging nicht, da WinForms nicht den Browser unterstützt. Es wird nach einem Update verlangt, welches nicht möglich ist.

## 7 Anforderung

Ich habe diese Anforderung fast geschafft. Ich habe eine Dll aus dem Internet heruntergeladen und diese danach mit einem Using verknüpft. Die Dll heisst System.Location und erlaubt es, dass man die aktuellen Koordinaten herauslest. Ich habe danach die URL der API erweitert und danach dem API übergeben. Die URL hat den Breitengrad und den Längengrad in sich. Ich empfang auch Sachen von der API, jedoch nicht meinen genauen Standort. Meine Vermutung ist, dass die Koordinaten nicht genau sind, bzw. sich auf das gesamte Land beziehen.   
Ich nehme die erste Station, welche zurückgegeben wird und gebe sie anschliessend in der TextBox meines User Controls aus.

## 8 Anforderung

Ich habe diese Anforderung nicht geschafft. Ich wollte es machen und habe auch meiner Meinung nach etwa den richtigen Code, jedoch klappt es nicht mit der Verbindung zum Server. Es kommt ein Fehler. Den Fehler habe ich abgefangen mit try-catch. Die Verbindungen hätte ich zu einem String konvertiert und danach als Text versendet.   
Die TextBoxen sind noch vorhanden, aber ich habe sie ausgeblendet. Den Quellcode habe ich auskommentiert.

# Unterprogramme

Ich habe zwei Unterprogramme geschrieben. Das erste heisst zeitFormatierer und formatiert die Ausgaben um. Man gibt das Objekt als String mit, welches formatiert werden soll und man gibt noch das Format mit, in welches formatiert werden soll. Das ganze wird mit einem tryparse() formatiert.

Ein weiteres Unterprogramm ist browserOeffner. Dieses Programm öffnet einen Browser mit Google Maps. Es wird die Station verlangt, von welcher man den Standort auf der Karte wissen will. Das Programm schaut, ob man etwas Leeres mitgibt, wenn nicht dann fügt es der Google Maps URL die Station hinzu.