

Dokumentation

ÖV-Applikation

21. Dezember 2018

Erik

Xingyuan

Cheng

ÜK M318

Analysieren und objektbasiert programmieren

Inhaltsverzeichnis

[1. Einleitung 2](#_Toc533170219)

[2. Anforderungen 2](#_Toc533170220)

[3. Zweck 3](#_Toc533170221)

[4. Richtlinien 3](#_Toc533170222)

[5. Analyse und design¨ 4](#_Toc533170223)

[6. Installation und Anwendung 8](#_Toc533170224)

[7. Testing 9](#_Toc533170225)

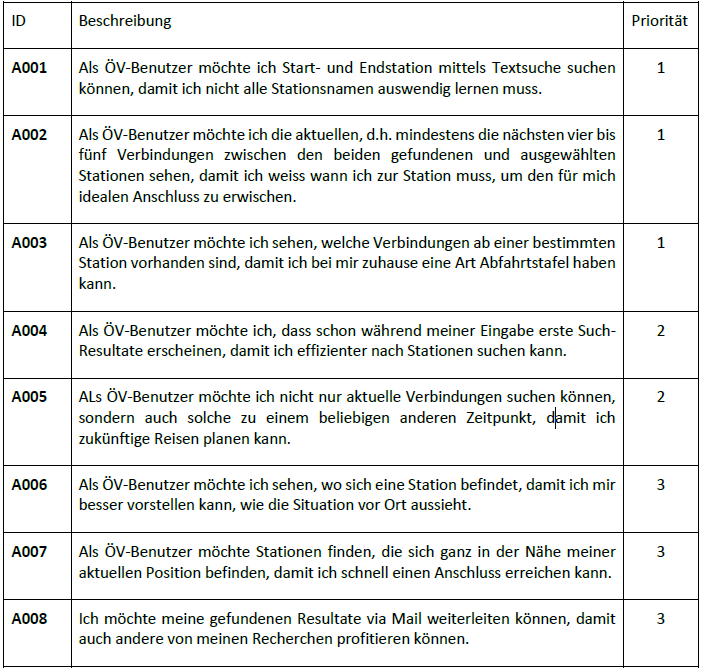
[8. Using Test 16](#_Toc533170226)

# Einleitung

Dieses Dokument ist da um zu erklären wie ich an meinem Projekt gearbeitet habe. Zu beschreiben welche Funktionen korrekt arbeiten oder, falls es bei einer noch Fehler gibt zu erklären warum es Fehler gibt. In diesem Dokument findet man auch alle Diagramme welche man benötigt um zu verstehen wie das Programm funktioniert. Auch werden hier die Testfälle und andere Informationen beschrieben.

# Anforderungen

In den vergangenen Woche bekam wir den Auftrag ein Programm mit C# zu implementieren. Das Programm musste mindestens folgende Funktionen beinhalten:



# Zweck

In diesem Dokument wird beschrieben, was das Programm macht und wie es gemacht wurde. Beilage wie Diagramme und andere zusätzliche sin auch angegeben. Installation und Deinstallation sind ebenfalls vorhandeln.

# Richtlinien

Naming Conventions

Lokale variablen beginnen mit einem Kleinbuchstaben, globale Variablen mit (\_).

Klassen und Methoden beginnen gross.Falls ein Name aus mehr als 2 Wörtern besteht wird jedes neue Wort grossgeschrieben.GUI Elemente beginnen mit einem Kürzel zB. cb für ComboBox oder lb für ListBox.Namen der Methoden müssen bereits die Aktion, welche sie ausführt beschreiben.

Deklaration

Variablen dürfen leer deklariert werden, müssen jedoch einen im Kontext eindeutigen Namen besitzen. Methoden dürfen lokale Variablen besitzen.

Kommentare

Jede public aufrufbare, nicht generierte Methode benötigt vorgehend einen Kommentar welcher den Sinn der Methode nochmals zusammenfasst. In der Methode fassen Kommentare einzelne Aktionen zusammen falls dies benötigt wird.

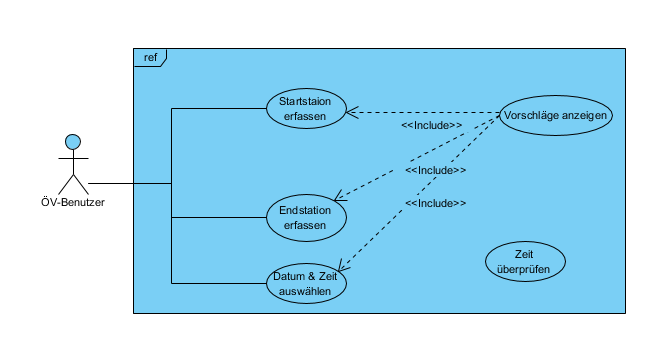
Statements

Falls ein Statement ausser Switch nicht bereits durch die Namensgebung der Variable logisch ist zB. if(nameVorhanden==true) wird vorgehend Kommentiert welche Bedingungen erfüllt erden werden müssen. Bei switch-Statements muss bei unklaren Optionen einen Kommentar vorgehen welcher die Aktion beschreibt oder ein allgemeiner Kommentar welcher das verhalten aller Optionen beschreibt. Falls die Anweisung nach einem Statement nur eine Zeile benötigt, werden die geschweiften Klammern weggelassen. Statements werden mit dem C#

# Analyse und design¨

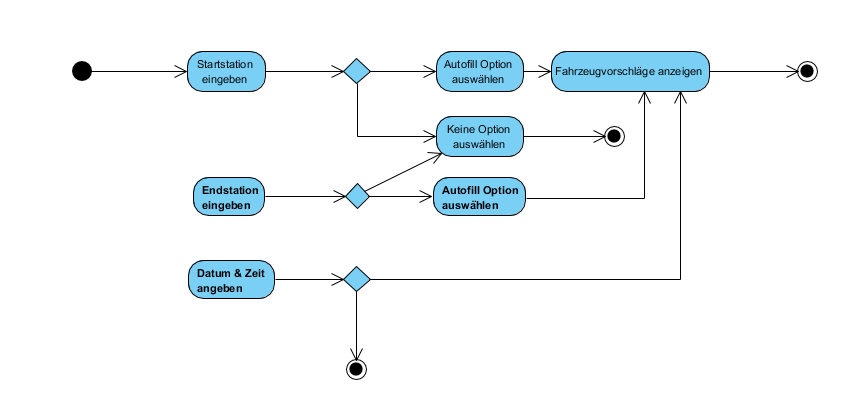
Use Case

Use Cases werden eingesetzt um die verschiedenen Funktionen aufzuzeigen, über die ein Programm verfügen muss. Man hat mindestens einen Actor (Aktor) und beliebig viele use cases(Anwendungsfälle), welche andere use cases includen(verwenden) können.



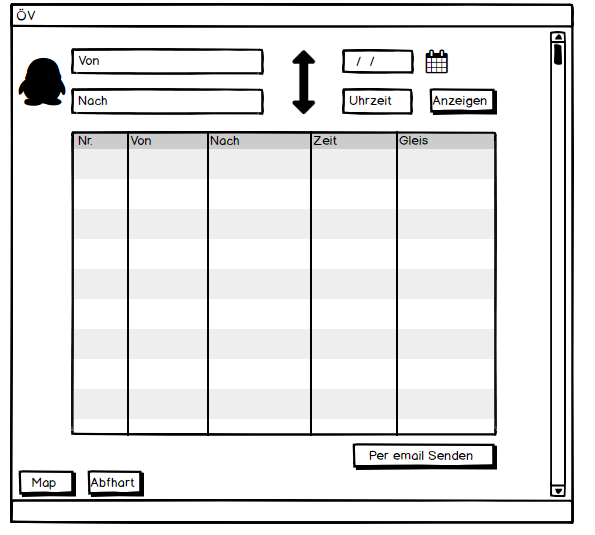
Aktivitätsdiagram

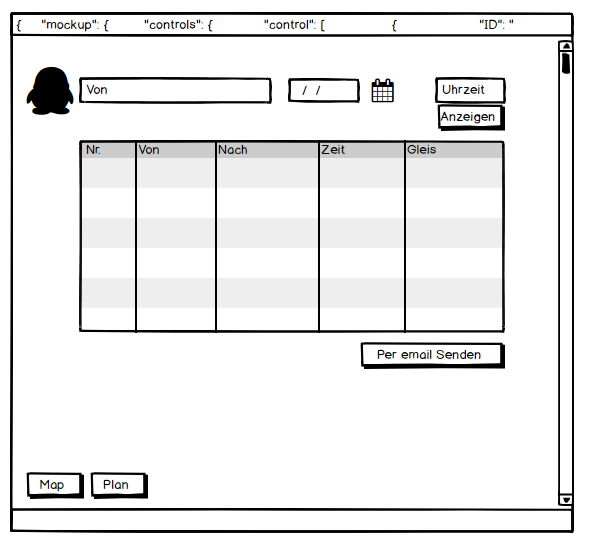
Das Aktivitätsdiagramm wird verwendet, um Daten- und Kontrollflüsse grafisch darzustellen. Es muss einen Startpunkt und einen oder mehrere Endpunkte haben. Dazwischen können verschiedene Aktivitäten stattfinden.

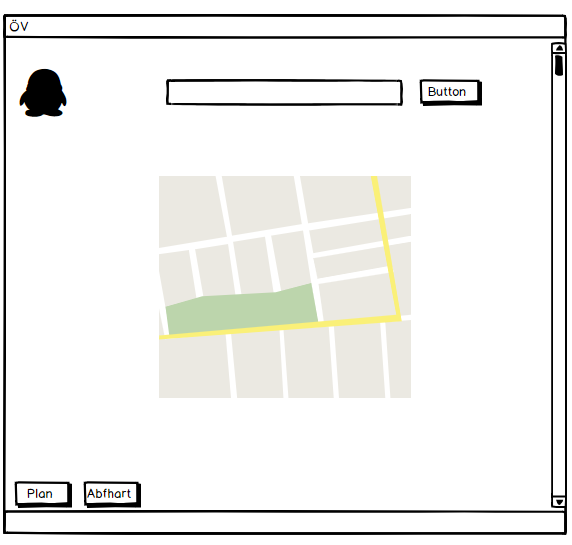


Mockup

Das Mockup ist das Design Die ich vor das Programmierung gemacht haben.

Erste Seite mit Verbindung Suchung.

Zweite Seite Mit Stationen Suchen.

Dritte Seite mit Google Maps.

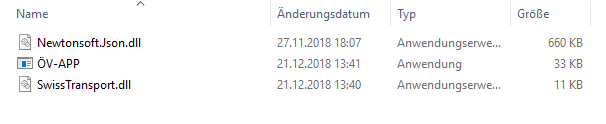
# Installation und Anwendung

Download Link: <https://github.com/aestusnero/modul-318-student.git>

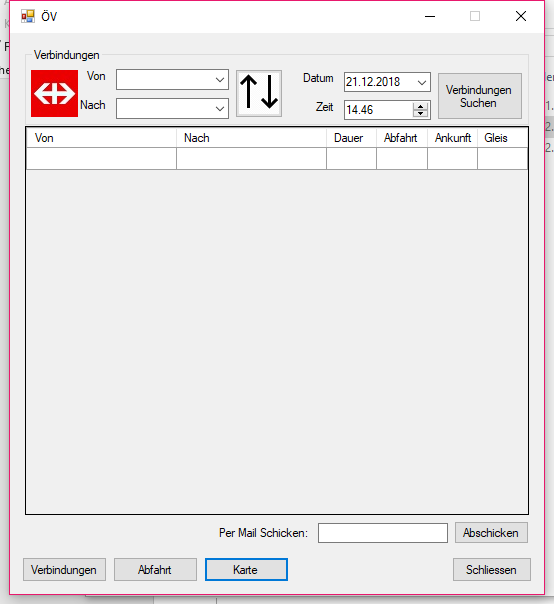
Schritt 1: Projekt herunterladen und aufmachen

Schritt 2: Navigieren zum APP Ordner und da ist das exe File.

Schritt 3: APP Aufmachen und benutzen. Aufpassen: das exe File und die zwei dll File müssen in gleiche Ordner sein!

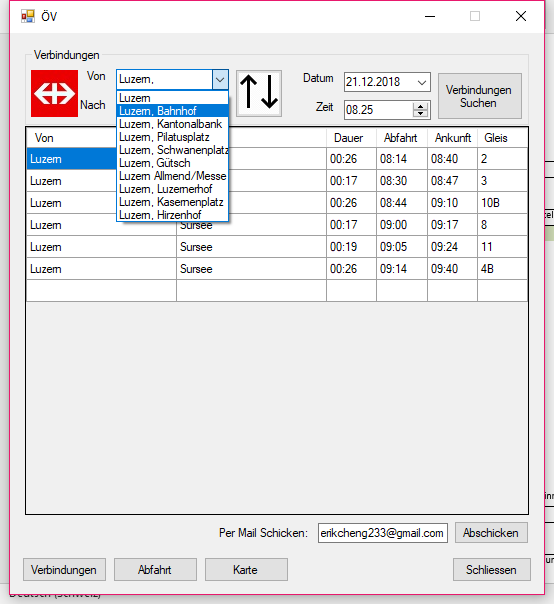


Bei Deinstallieren einfach löschen.

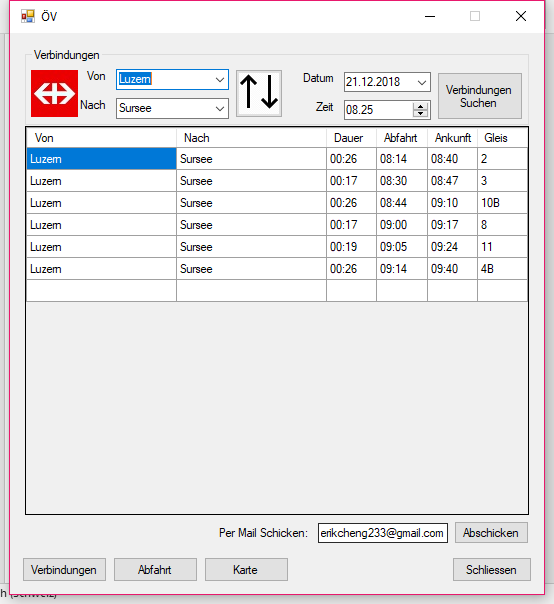


# Testing

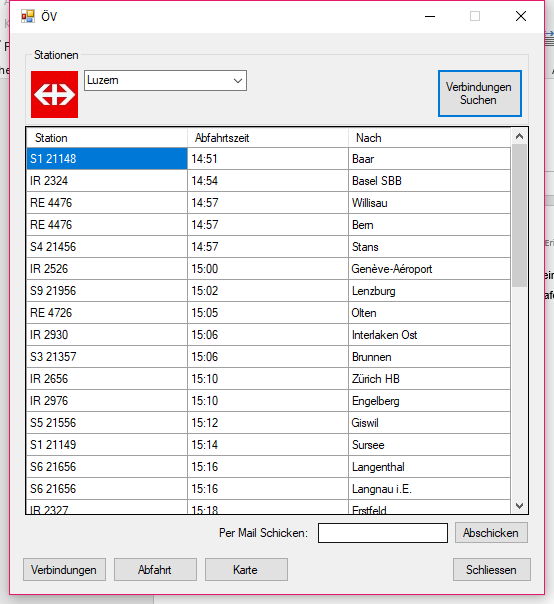
A001 und 4: Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss.



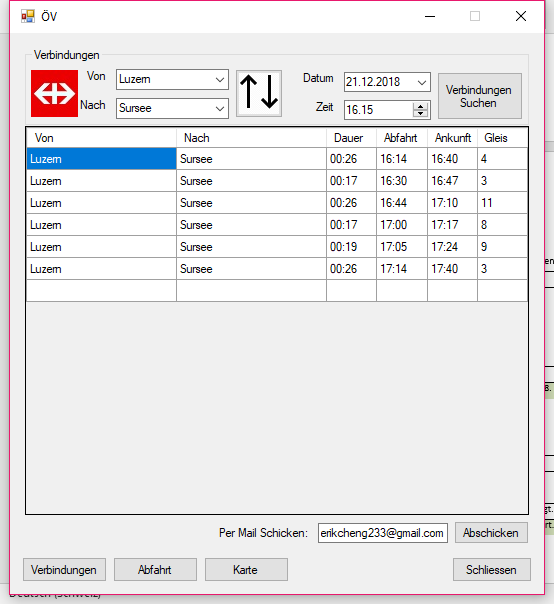
A002: Als ÖV-Benutzer möchte ich die aktuellen, d.h. mindestens die nächsten vier bis fünf Verbindungen zwischen den beiden gefundenen und ausgewählten Stationen sehen, damit ich weiss wann ich zur Station muss, um den für mich idealen Anschluss zu erwischen.



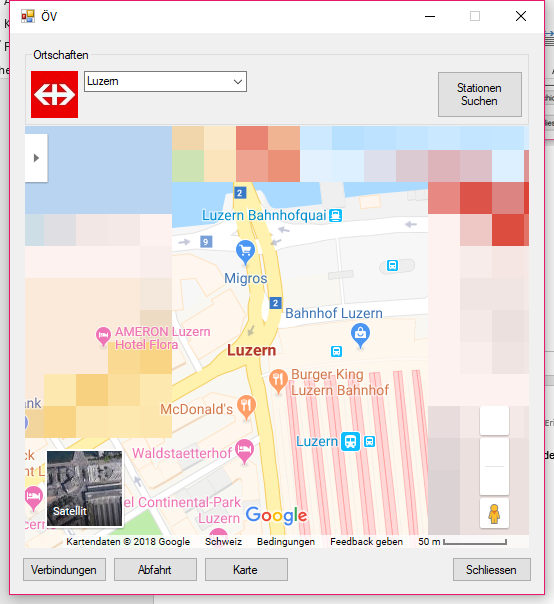
A003: Als ÖV-Benutzer möchte ich sehen, welche Verbindungen ab einer bestimmten Station vorhanden sind, damit ich bei mir zuhause eine Art Abfahrtstafel haben kann.



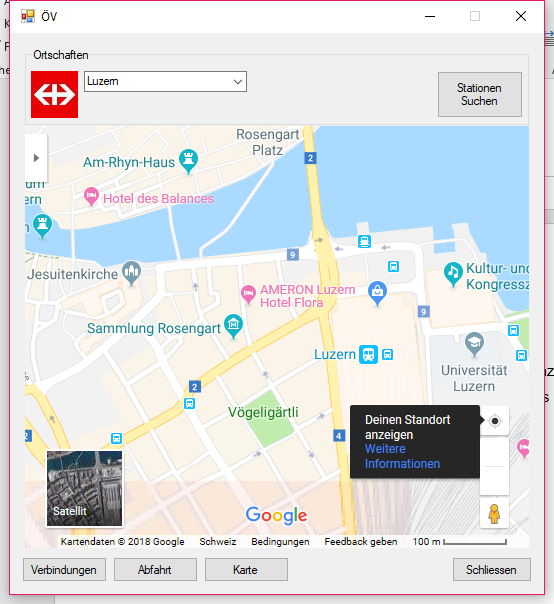
A005: ALs ÖV-Benutzer möchte ich nicht nur aktuelle Verbindungen suchen können, sondern auch solche zu einem beliebigen anderen Zeitpunkt, damit ich zukünftige Reisen planen kann.



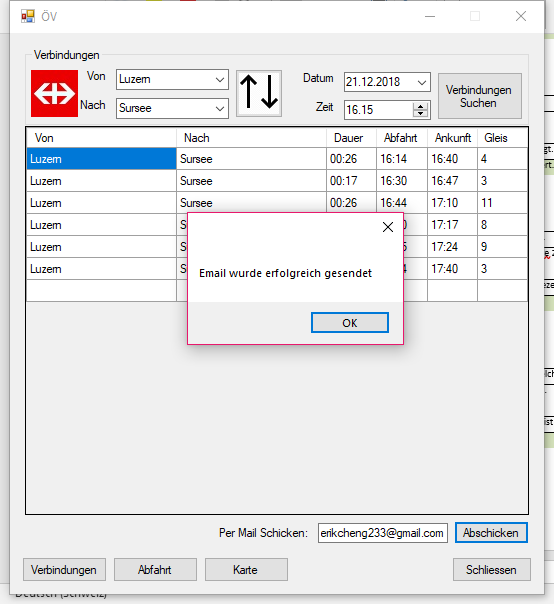
A006: Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht.



A007: Als ÖV-Benutzer möchte Stationen finden, die sich ganz in der Nähe meiner aktuellen Position befinden, damit ich schnell einen Anschluss erreichen kann.

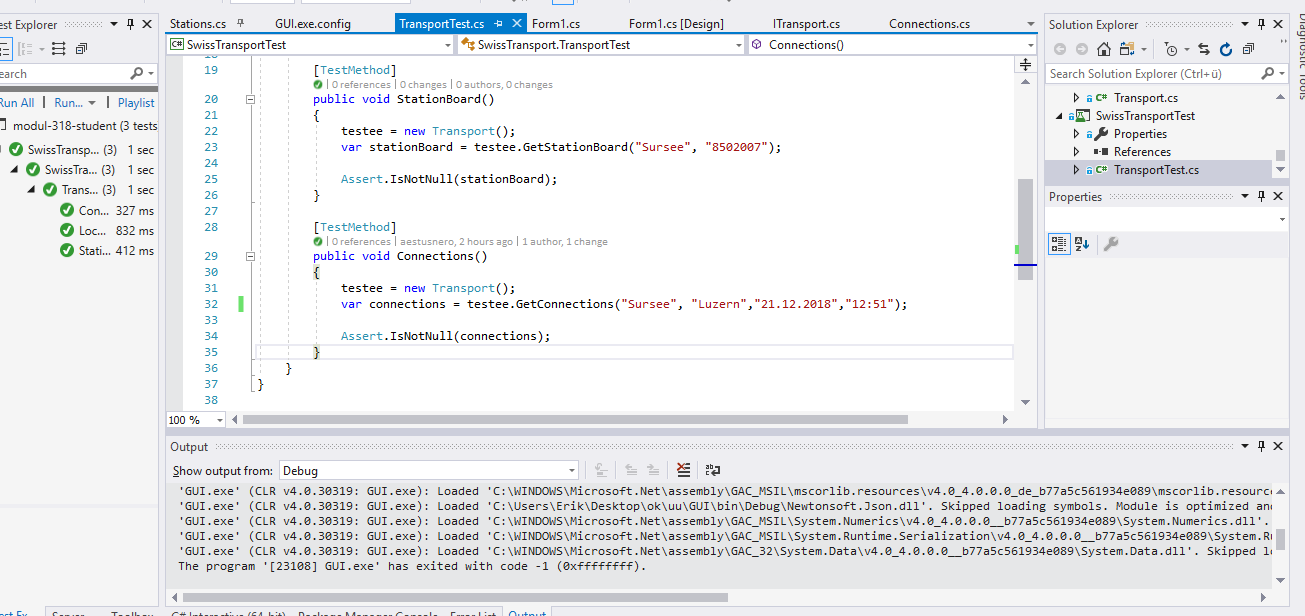


A008: Ich möchte meine gefundenen Resultate via Mail weiterleiten können, damit auch andere von meinen Recherchen profitieren können.





# Using Test



public void Connections()

{

testee = new Transport();

var connections = testee.GetConnections("Sursee","Luzern","21.12.2018","12:51");

Assert.IsNotNull(connections);

}

Ich musste dabei Datum und Zeit in ITransport ändern.

