M318 Dokumentation David Peric

David Peric, 10.12.2019 – 19.12.2019

Inhalt

1 Management Summary

Im Rahmen des ÜK M318 habe ich das Projekt «FahrplanApp» entwickelt. Das Ziel des Projektes war es, die Kenntnisse in der OOP zu verbessern und einen Einblick in die Entwicklung einer Applikation (mit Planung, etc.) zu erhalten. Dabei war es auch wichtig die Verwendung von HTTP Web API's zu erlernen.

Ob diese Ziele erreicht wurden, lässt sich in der Dokumentation herausfinden.

2 Zweck

Der Zweck, bzw. der Sinn der Dokumentation ist es, das Programm «FahrplanApp» dem Leser näher zu bringen und die Planung, sowie Umsetzung des Projektes anhand der Web API zu dokumentieren. Dieses Projekt wurde im Rahmen des ÜKs 318 (Objektorientiertes Programmieren) umgesetzt und wurde von mir entwickelt.

Dabei ist der Sinn und Zweck der Übung die Kenntnisse im OOP zu verbessern.

3 Analyse

3.1 Funktionalität

Im Projekt geht es darum, mindestens die ersten drei der acht Features umzusetzen. Diese Features werden im Kapitel Funktionalitäten beschrieben.

3.2 Qualitätsmerkmale

1. **Funktionalität** Hier wird geprüft, ob die Software richtig funktioniert.
2. **Zuverlässigkeit** Hier schaut man ob die Software zuverlässig funktioniert, was in Fehler-Situationen passiert, ob verständliche Fehlermeldungen angezeigt werden, ob sich Daten wiederherstellen lassen usw.
3. **Benutzbarkeit** Kann man die Software einfach bedienen? Ist die Benutzeroberfläche übersichtlich und selbsterklärend gestaltet? Versteht man die Funktionen?
4. **Effizienz** Definiert die Performance (das Zeitverhalten) der Applikation, den Speicherverbrauch usw.
5. **Wartbarkeit** Kann man die Software einfach/problemlos erweitern? Wie einfach ist es, Fehler einzugrenzen und zu lokalisieren? Wie stabil verhält sich die Software bei Änderungen (Stichwort: Side Effects)? Ist der Source Code gut und verständlich kommentiert? Wurden die Programmierrichtlinien durchgängig eingehalten?
6. **Portierbarkeit** Kann man die Software einfach installieren? Kann sie mit anderen Programmen koexistieren? Welche Auswirkungen hat die Installation der Software auf andere Programme? usw.

3.3 Qualitätsziele

Funktionalität:
- Die Funktionalität der Software soll den Anforderungen entsprechen, welche laut Dokumentation umgesetzt wurden.
- Die Funktionen sollen korrekt umgesetzt sein.

- Die richtige Funktionsweise soll durch Testfälle sichergestellt sein.

Zuverlässigkeit:
- Die Software soll zuverlässig funktionieren und tun, was man von ihr erwartet.
- Die Software soll in Fehlerfällen den Benutzer korrekt (mit verständlichen Fehlermeldungen) informieren ohne abzustürzen.
- Nach Auftreten eines Fehlers sollte die Software, wenn möglich, normal weiterlaufen.

Benutzbarkeit:
- Die Bedienung der Software soll selbsterklärend sein.
- Benutzereingaben sollen validiert werden.
- Die grafische Benutzeroberfläche (GUI) soll übersichtlich gestaltet sein.
- Das GUI soll sich an UI Standards halten.

Wartbarkeit:
- Der Source Code soll sich an die Programmierrichtlinien (Coding Guidelines) halten.
- Variablen, Klassen, Methoden und andere Elemente sollen sprechende Namen haben.
- Der Source Code soll verständlich kommentiert sein, insbesondere sollen alle öffentlichen Methoden inklusive ihrer Parameter kommentiert sein.
- Es soll einfach und problemlos möglich sein, Änderungen an der Software zu machen. Dazu gehört: kein repetitiver Code (copy paste), kein Spaghetticode, Beachtung des Kohäsionszprinzips und eine möglichst lose Kopplung der einzelnen Module und Klassen.

Portierbarkeit:
- Die Software soll einfach zu installieren sein.
- Die Software soll auch einfach wieder entfernt werden können.
- Die Software soll nach der Installation auch auf einem Rechner ohne Entwicklungsumgebung lauffähig sein.
- Zusammen mit der Software soll eine Installationsanleitung ausgeliefert werden.

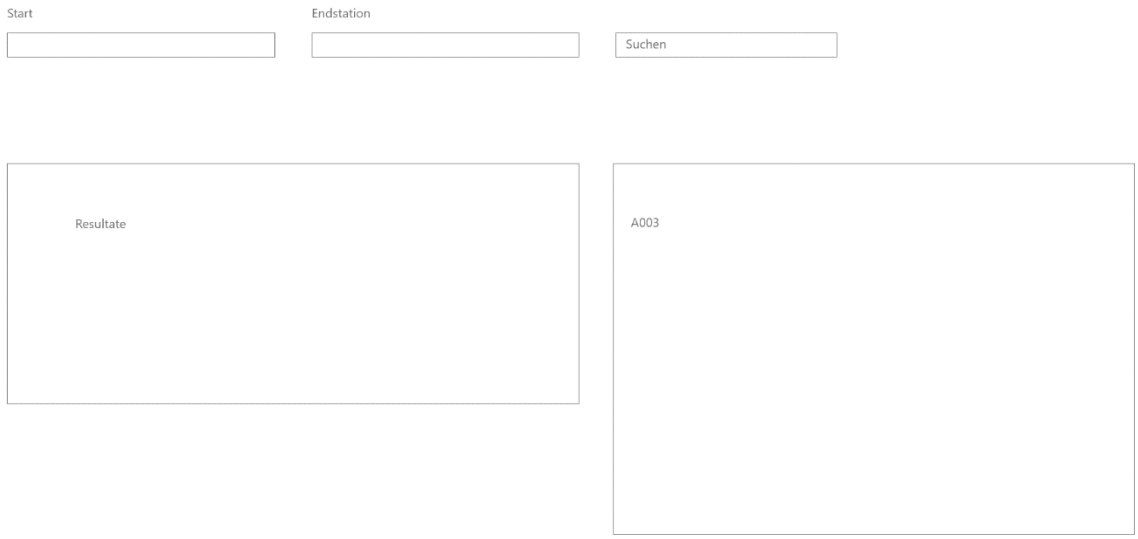
3.4 Dokumentation

Die Dokumentation soll nach folgenden Kriterien bewertet werden:

- Autor, Datum
- Eine Einleitung (Management Summary).
- Zweck des Dokuments.
- Was (d.h. welche Funktionen) wurde umgesetzt?
- Falls bestimmte Funktionen nur teilweise umgesetzt wurden: Welche? Welcher Teil der Funktionalität fehlt noch? Bekannte Fehler/Bugs?
- Use Cases und Aktivitätendiagramme für alle umgesetzten Anforderungen aber mindestens für die mit Priorität 1 klassifizierten Anforderungen.
- Testfälle (Systemtests), verständlich und **eindeutig nachvollziehbar** geschrieben, so dass ein Tester diese ohne weiteren Erklärungen durchführen kann.
- Installationsanleitung: Wie wird die Software installiert? Wie wird die Software deinstalliert?
- Andere spannende Informationen für die Bewertung.
- Inhaltsverzeichnis vorhanden

4 UI Mockups

4.1 Mockup



4.2 Unterschiede zum Endprodukt

Das Mockup unterscheidet sich stark zum Endprodukt. Beim Endprodukt habe ich mich entschieden Tabs zu brauchen. So kam es dazu, dass der erste Tab dazu dient Verbindungen zu suchen, der zweite, um eine Abfahrtstafel anzuzeigen und der dritte um die Stationen auf der Karte anzuzeigen.

5 Funktionalitäten

Das Projekt bestand darin, folgende Anforderungen für das Programm zu erfüllen:

A001	Als ÖV-Benutzer möchte ich Start- und Endstation mittels Textsuche suchen können, damit ich nicht alle Stationsnamen auswendig lernen muss.	Erledigt	1
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.	Erledigt	1
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.	Erledigt	1
A004	Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Such-Resultate erscheinen, damit ich effizienter nach Stationen suchen kann.	Erledigt	2
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.	Erledigt	2
A006	Als ÖV-Benutzer möchte ich sehen, wo sich eine Station befindet, damit ich mir besser vorstellen kann, wie die Situation vor Ort aussieht.	Erledigt	3
A007	Als ÖV-Benutzer möchte Stationen finden, die sich ganz in der Nähe meiner aktuellen Position befinden, damit ich schnell einen Anschluss erreichen kann.	Nicht erledigt	3
A008	Ich möchte meine gefundenen Resultate via Mail weiterleiten können, damit auch andere von meinen Recherchen profitieren können.	Nicht erledigt	3

Dabei gilt folgende Skala:
1. Must
2. Should
3. Nice to have

In diesem Projekt wurden alle Features ausser A007 und A008 implementiert.

5.1 A001
Die Start- und Endstation können innerhalb des Programmes per Text in der TextBox per TextSuche gefiltert werden.

5.2 A002
Die nächsten vier bzw. fünf Verbindungen zwischen den zwei ausgewählten Stationen werden in einer ListView dargestellt.

5.3 A003
Die Abfahrtstafel kann für die ausgewählte Station angezeigt werden. Folgende Informationen werden angezeigt:

- Betreiber
- Abfahrtszeit
- Richtung

5.4 A004
Während dem Tippen erscheinen Vorschläge, bzw. Resultate in einer ListBox. Die ListBox wird nur angezeigt, wenn sie etwas drinnen hat, bzw. nicht leer ist. Nach dem man eine Station, bzw. ein Element aus der ListBox ausgewählt hat, wird sie nicht mehr angezeigt.

5.5 A005
Durch einen Klick auf «Mehr Optionen anzeigen» können Verbindungen zu einem beliebigen Zeitpunkt ausgewählt werden.

5.6 A006
Im Tab Station kann nach Stationen gesucht werden. Nach dem man den Button drückt oder Enter drückt, werden die Stationen in einer ListBox angezeigt. Wenn man dann auf eine Station doppelklickt, wird die Karte abgerufen. Der Karte werden die Koordinaten übergeben, jedoch werden diese nicht im Programm angezeigt.

5.7 A007
Dieses Feature habe ich nicht implementiert aufgrund von Zeitmangel und des grossen Aufwandes.

5.8 A007
Dieses Feature habe ich ebenfalls aus den gleichen Gründen wie bei 5.7 nicht implementiert.

6 Code Style und Quality
In diesem Projekt wurden folgende Code Quality Massnahmen angewendet und folgende Code Style Programmierrichtlinien verwendet.

6.1 Style und Quality
- Methoden und Namen von Steuerelementen sind in CamelCase geschrieben
- Auf Deutsch kommentiert

- Kommentare immer auf eine Linie schreiben
- Steuerelemente sinnvoll benannt

7 Programmbeschrieb

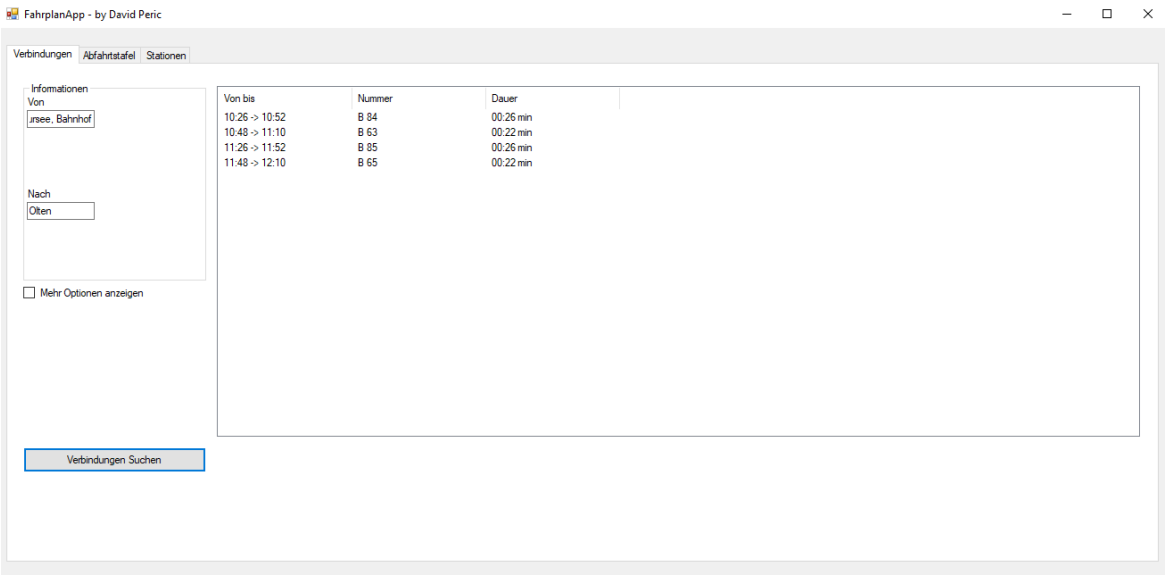
Das Programm «FahrplanApp» ist unterteilt in drei Funktionalitätsgruppen:

- Verbindungen
- Abfahrtstafel
- Stationen

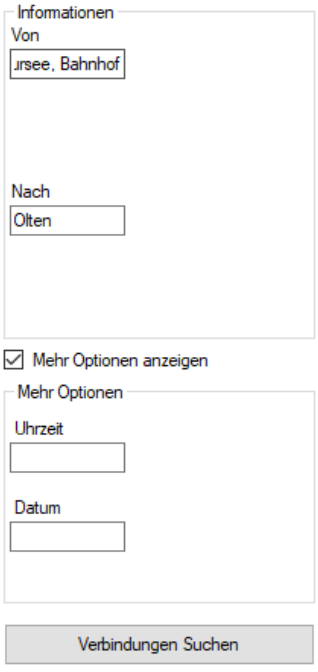
Jede dieser Funktionalitätsgruppen wird nachfolgend genauer beschrieben.

7.1 Verbindungen

Nachfolgend ist ein Screenshot vom Tab «Verbindungen» zu sehen.



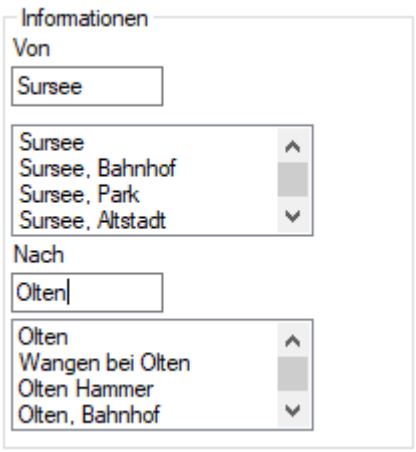
Diese Funktionalitätsgruppe ist dabei weiter aufgeteilt in die linke und rechte Spalte. In der linken Spalte befinden sich alle Optionen auf der rechten Seite alle Informationen, bzw. Resultate.



Durch einen Klick auf «Mehr Optionen anzeigen» weitere Optionen in der linken Spalte eingeblendet. Diese sind standardmässig nicht eingeschaltet, da sie nur in Sonderfällen verwendet werden und ansonsten den Nutzer überfordern würden.

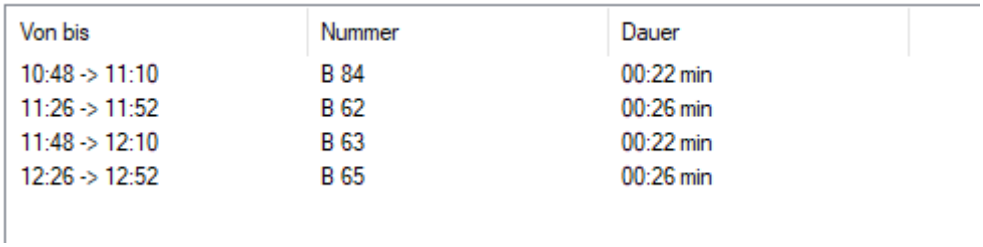
In der oberen GroupBox können die Stationen ausgewählt werden. Während man tippt, werden Vorschläge, bzw. Resultate angezeigt in einer ListBox. Die ListBox ist standardmässig nicht sichtbar, sondern wird erst sichtbar, wenn sie mit Daten gefüllt ist, bzw. nicht leer ist. Wenn man ein Element aus der ListBox mit Doppelklick auswählt, wird die TextBox, mit der Station, bzw. den Daten des ausgewählten Elements gefüllt.

Wenn die Option «Mehr Optionen anzeigen» deaktiviert ist, wird als Datum und Uhrzeit für die Abfrage, bzw. Suche das aktuelle Datum und die aktuelle Uhrzeit verwendet. Dies gilt generell, falls eines der Felder in der GroupBox «Mehr Optionen» leer ist.



Hier wird gezeigt, was passiert, wenn man am Tippen ist. Nach dem man eine Station auswählt wird die ListBox wieder nicht sichtbar.

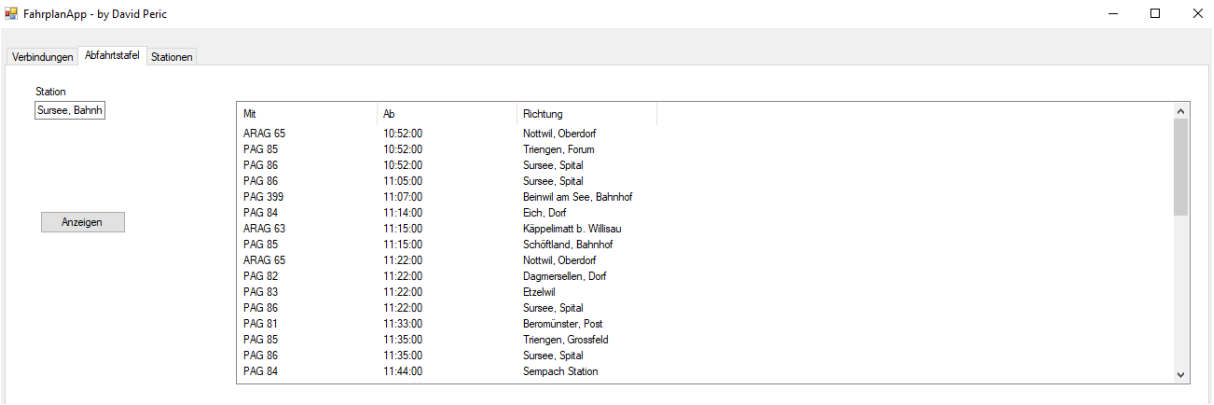
Durch einen Klick auf «Verbindungen suchen» werden die Informationen, bzw. Resultate in der ListView auf der rechten Seite angezeigt. Wenn die zwei Felder in der GroupBox oben leer sind, wird eine Meldung ausgegeben, dass es Pflichtfelder sind.



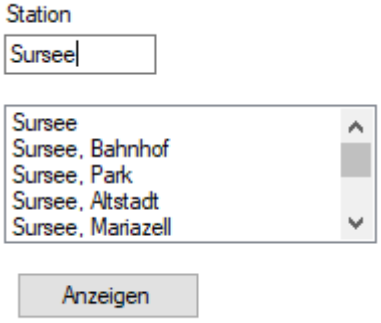
Von bis	Nummer	Dauer
10:48 -> 11:10	B 84	00:22 min
11:26 -> 11:52	B 62	00:26 min
11:48 -> 12:10	B 63	00:22 min
12:26 -> 12:52	B 65	00:26 min

7.2 Abfahrtstafel

Die Abfahrtstafel zeigt alle Verbindungen ab der ausgewählten Station an.



Hier wird ebenfalls zwischen zwei Spalten unterschieden: Optionen und Informationen. In der linken Spalte muss man nur die Station auswählen. Während dem Tippen werden Vorschläge, bzw. Resultate in einer ListBox angezeigt. Standardmässig ist die ListBox nicht sichtbar, sondern wird erst sichtbar, wenn sie mit Daten gefüllt ist, bzw. nicht leer ist. Wenn man ein Element, bzw. eine Station aus der ListBox auswählt, werden in der ListView die Resultate angezeigt.



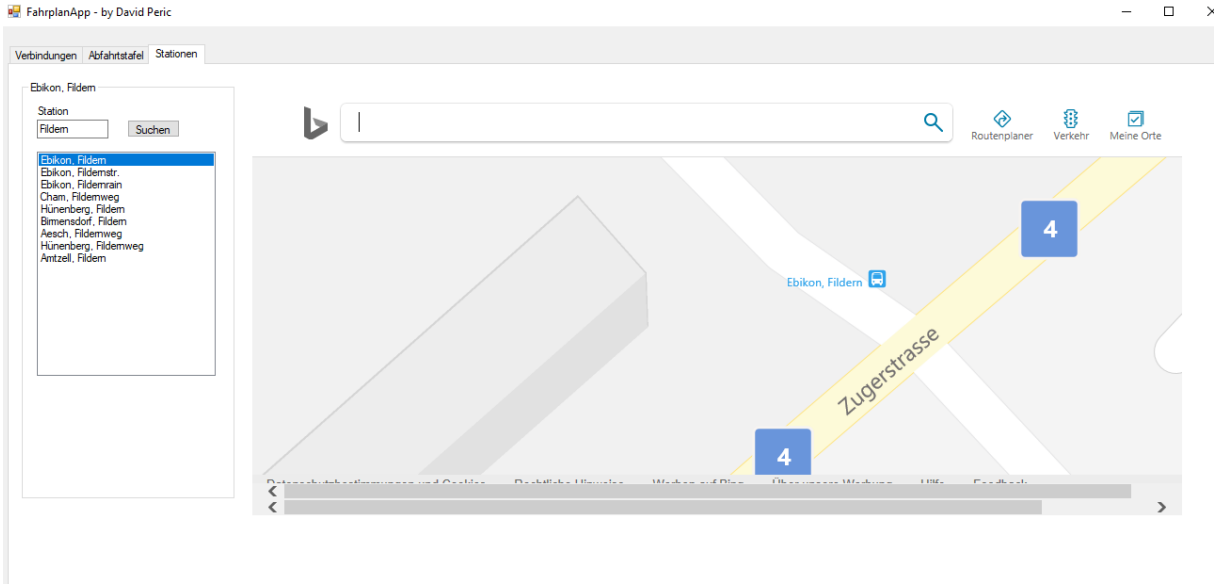
Hier wird gezeigt, was passiert, wenn man am Tippen ist. Nachdem man eine Station ausgewählt hat, wird die ListBox wieder nicht sichtbar.

Durch einen Klick auf «Anzeigen», werden die Resultate in der ListView auf der rechten Seite angezeigt. Alternativ zum Buttonklick kann man auch den Button «Anzeigen» auslösen durch das Drücken von der Taste Enter oder durch Doppelklick auf eine Station. Wenn man keine Station eingibt

oder auswählt jedoch «Anzeigen» oder Enter drückt, bekommt man die Fehlermeldung, dass es ein Pflichtfeld ist.

Mit	Ab	Richtung
PAG 86	11:05:00	Sursee, Spital
PAG 399	11:07:00	Beinwil am See, Bahnhof
PAG 84	11:14:00	Eich, Dorf
ARAG 63	11:15:00	Käppelimatt b. Willisau
PAG 85	11:15:00	Schöftland, Bahnhof
ARAG 65	11:22:00	Nottwil, Oberdorf
PAG 82	11:22:00	Dagmersellen, Dorf
PAG 83	11:22:00	Etzelwil
PAG 86	11:22:00	Sursee, Spital
PAG 81	11:33:00	Beromünster, Post
PAG 85	11:35:00	Triengen, Grossfeld
PAG 86	11:35:00	Sursee, Spital
PAG 84	11:44:00	Sempach Station
ARAG 62	11:45:00	Ruswil, Rottalcenter
ARAG 63	11:45:00	Käppelimatt b. Willisau
ARAG 65	11:52:00	Nottwil, Oberdorf

7.3 Stationen



Im Tab Stationen, kann man nach Stationen suchen. Hier ist es ebenfalls in zwei Spalten aufgeteilt. In der linken Spalte werden Stationen gesucht und anschliessend ausgewählt und auf der rechten Spalte werden im WebBrowser die Informationen angezeigt. Wenn die TextBox leer ist, und man auf den Button «Suchen» klickt oder Enter drückt, bekommt man eine Fehlermeldung, dass es sich dabei um ein Pflichtfeld handelt.

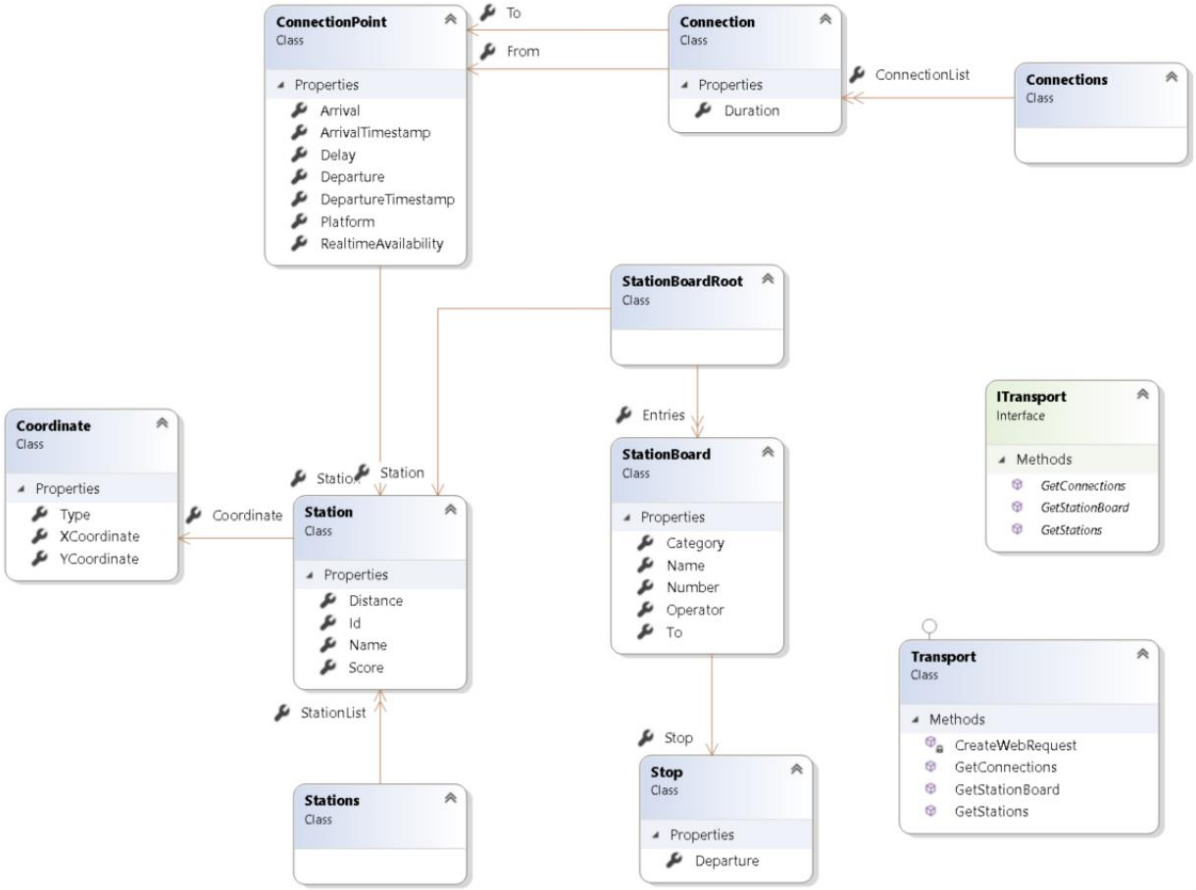
Wenn man einen Ort, bzw. eine Station mit Doppelklick auswählt, werden die Koordinaten zu diesem Ort abgerufen und dann am Programm übergeben. Die Koordinaten werden jedoch nicht im UI angezeigt für den User. Wenn die Koordinaten vom ausgewählten Ort den Wert «0» haben, bekommt man eine Fehlermeldung, dass man einen anderen Ort auswählen muss.

8 Project Dependencies

Damit man die API im eigenen Projekt brauchen kann, muss man einen Verweis zur Projektmappe von der API hinzufügen. Damit das richtige kompiliert, muss man das eigene Projekt als Startprojekt festlegen.

9 Klassendiagramme

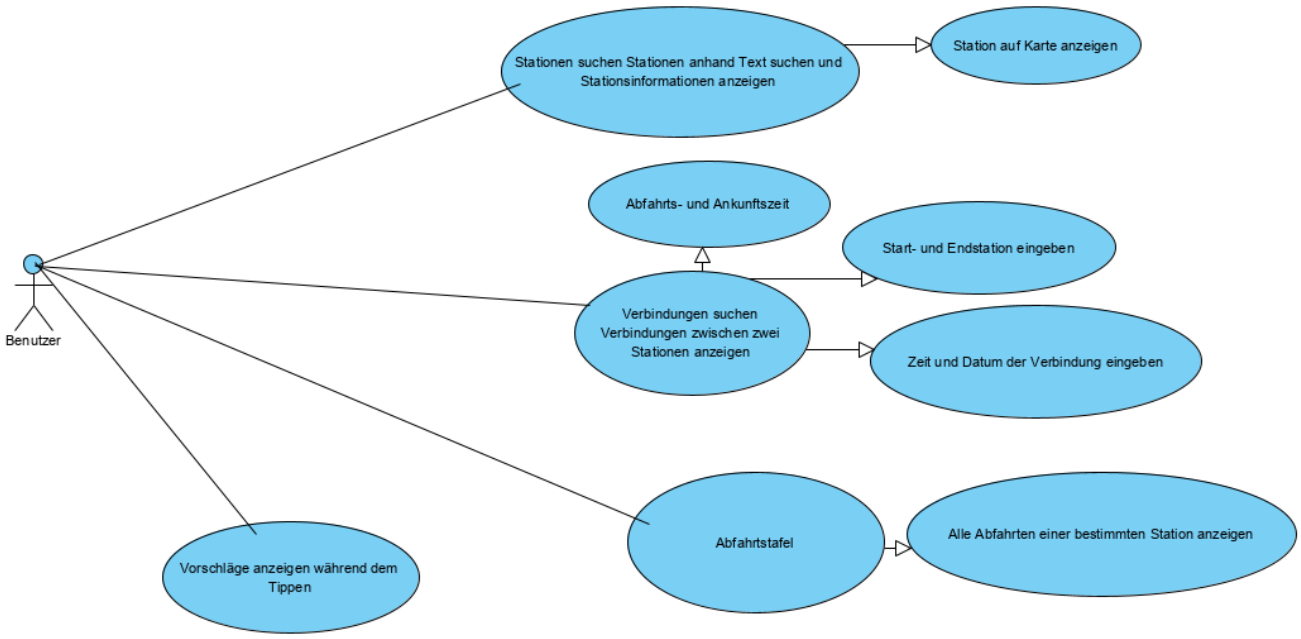
9.1 SwissTransport



10 Use Cases

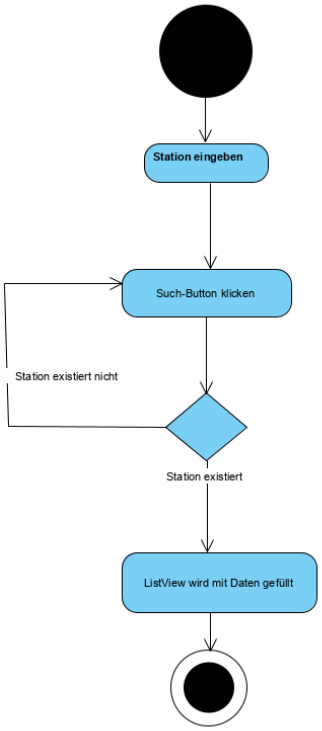
Im folgenden Diagramm wurden alle UseCases meiner Applikation «FahrplanApp» übersichtlich dargestellt.

10.1 Diagramm

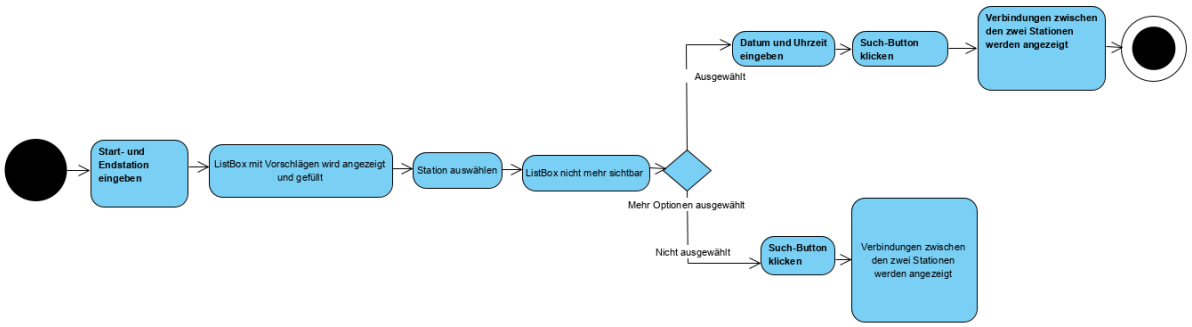


11 Aktivitätendiagramme

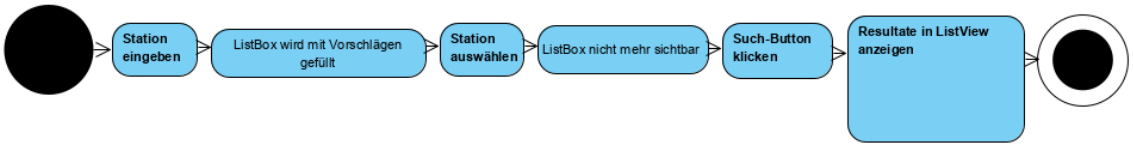
11.1 A001



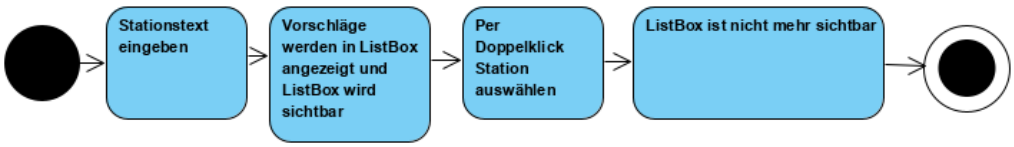
11.2 A002



11.3 A003



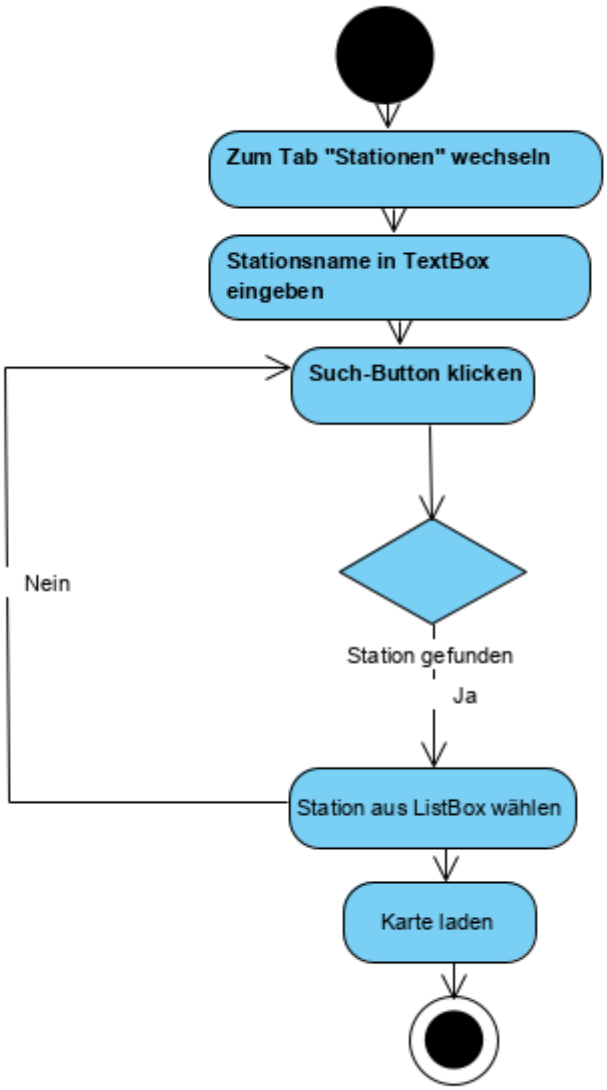
11.4 A004



11.5 A005

Siehe A002

11.6 A006



12 Systemtests

12.1 Voraussetzungen

Für alle Tests muss der verwendete Computer eine aktive Internetverbindung haben, da sonst keine Kommunikation zur API möglich ist. Die Applikation muss über das .exe File gestartet werden.

12.2 Verbindungen suchen

Name	**Erwartetes Resultat**
1. Zum Tab «Verbindungen» wechseln 2. Eine beliebige Start- und Endstation eingeben	Die ListBox wird sichtbar und Vorschläge werden angezeigt.
3. Start- und Endstation auswählen	Nach dem man mit Doppelklick die Stationen auswählt verschwindet die ListBox und ist nicht mehr sichtbar.

4. «Mehr Optionen anzeigen» klicken 5. Datum und Uhrzeit auswählen, bzw. eingeben	Die GroupBox mit den weiteren Optionen ist sichtbar.
6. «Verbindungen suchen» klicken	Die Verbindungen werden in der ListView angezeigt. Folgende Eigenschaften werden angezeigt: - Von bis (Abfahrt/Ankuft) - Zug- oder Busnummer - Dauer

12.3 Abfahrtstafel

Name	**Erwartetes Resultat**
1. Zum Tab «Abfahrtstafel» wechseln	Der Tab wird angezeigt
2. Station eingeben	ListBox mit Vorschlägen wird angezeigt.
3. Station auswählen	ListBox ist nicht mehr sichtbar und Stationen werden in ListView angezeigt.
4. «Anzeigen» klicken (optional)	Falls geklickt, passiert das gleiche, wie wenn man eine Station auswählt.

12.4 Stationen suchen

Name	**Erwartetes Resultat**
1. Zum Tab «Stationen» wechseln	Der Tab wird angezeigt.
2. Name der gewünschten Station eingeben 3. «Suchen» klicken	Nach dem klicken von «Suchen» werden alle Resultate in einer ListBox angezeigt.
4. Station auswählen	- Der Text von der GroupBox wird zu dem Wert von der ausgewählten Station geändert. - Karte wird im WebBrowser geladen, anhand von den Koordinaten

12.5 Invalide Daten

Name	**Erwartetes Resultat**
1. Stationsname leer lassen	
2. Button klicken (z.B. «Anzeigen»	Eine Fehlermeldung wird angezeigt und wartet darauf, dass man sie schliesst.

13 Erkannte Fehler

Mir ist aufgefallen, dass es bei der API einen Fehler gibt, dass wenn man zu schnell tippt, dass das Programm abstürzt. Ich habe auch nicht herausgefunden, wie dass man den Fehler abfängt. Der selbe Fehler kann auch passieren, wenn man den Ort aus versehen falsch eingibt.

14 Deployment

Die Software ist auf meinem Git-Repository runterladbar.

14.1 Installation

Zur Installation muss man das Git-Repo runterladen und die .exe Datei im Debug Ordner öffnen.

14.2 Deinstallation

Zum Deinstallieren, einfach das runtergeladene Git-Repo vom PC/Computer löschen.

15 Reflexion

Ich habe in diesem Projekt, bzw. ÜK viel über C# Events und Steuerelemente gelernt. Ich hätte aber von Anfang an, den Steuerelementen sinnvolle Namen geben müssen, weil das zu ändern war recht «nervig» und eigentlich auch Zeitverschwendung. Den Code hätte ich von Anfang an kommentieren sollen. Jedoch habe ich vieles wichtiges gelernt, wie z.B. wie das man den WebBrowser in eine C# App einbindet. Am wichtigsten war für mich aber, dass ich in diesem ÜK gesehen habe, dass man mit C# auch spannende Sachen erstellen kann und nicht nur so «Statische Apps» wie in der Schule.