SBB Station Finder

ÜK Projekt

Patrick Schneeberger

Inhaltsverzeichnis

[1.0 Vorwort 3](#_Toc34921097)

[2.0 Installation 3](#_Toc34921098)

[3.0 Anforderungen 4](#_Toc34921099)

[3.1 Kundenanforderungen 4](#_Toc34921100)

[4.0 Grafische Darstellung 5](#_Toc34921101)

[4.1 Mokup - frmMain 5](#_Toc34921102)

[4.2 Mokup - frmInfo 5](#_Toc34921103)

[5.0 Use Case 6](#_Toc34921104)

[5.1 Diagramm 6](#_Toc34921105)

[5.2 Beschreibung - Stationen eingeben 6](#_Toc34921106)

[5.3 Beschreibung - Zeitpunkt eingeben 7](#_Toc34921107)

[5.4 Beschreibung - Fahrplanverbindungen anzeigen 7](#_Toc34921108)

[6.0 Programmierrichtlinien 8](#_Toc34921109)

[6.1 Benennung der visuellen Elemente 8](#_Toc34921110)

[6.2 Codierrichtlinien 8](#_Toc34921111)

[7.0 Umgesetzte Anforderungen 9](#_Toc34921112)

[7.1 Bekannte Bugs 9](#_Toc34921113)

[8.0 Manueller Test 10](#_Toc34921114)

[8.1 Use Case 1 - Startstation, Endstation eingeben 10](#_Toc34921115)

[8.2 Use Case 2 - Zeitpunkt eingeben 10](#_Toc34921116)

[8.3 Use Case 3 - Fahrplanverbindungen anzeigen 11](#_Toc34921117)

[8.4 Erweiterte Tests 11](#_Toc34921118)

[9.0 Kürzübersicht über eigene Methoden und Funktionen 12](#_Toc34921119)

[9.1 frmMain.cs 12](#_Toc34921120)

[9.2 frmMail.cs 12](#_Toc34921121)

[10.0 API Erweiterung 12](#_Toc34921122)

1. Vorwort

Dies ist die Dokumentation zur Applikation "SBB Station Finder" den wir im ÜK-318 Programmiert haben. Das Ziel dieses Projekts war es eine Applikation zu schreiben die eine API abfragt. Die API liefert Informationen zum SBB Fahrplan der Schweiz.

Das Programm wird mit Visual Studio in C# geschrieben. Die Applikation ist eine Windows Forms-App(.NET Framework 4.7.2).

1. Installation

Installation

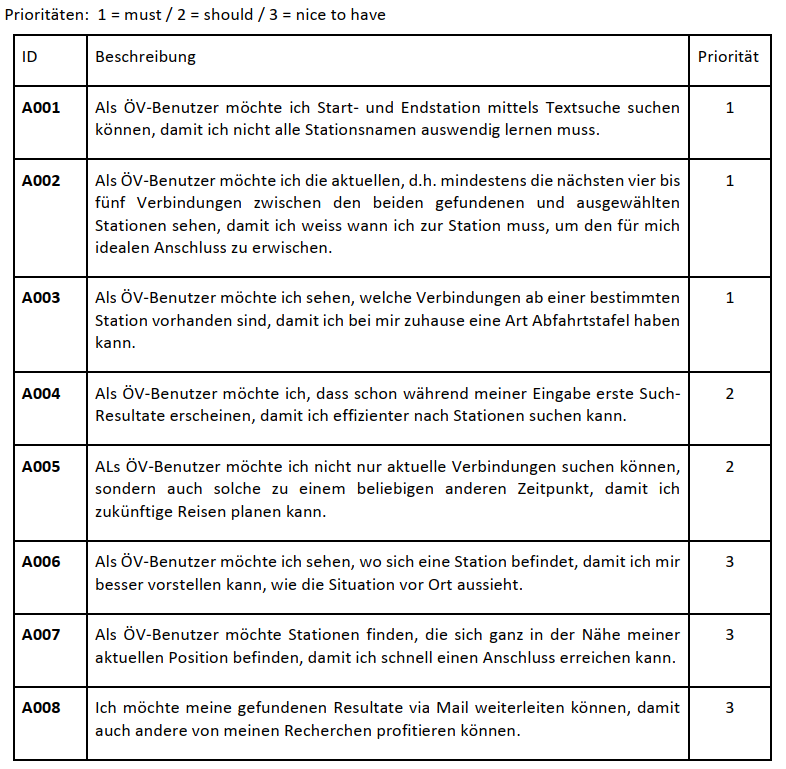
1) StationFinderSFX.exe ausführen

2) Optional: Pfad anpassen

3) auf "Extract" klicken

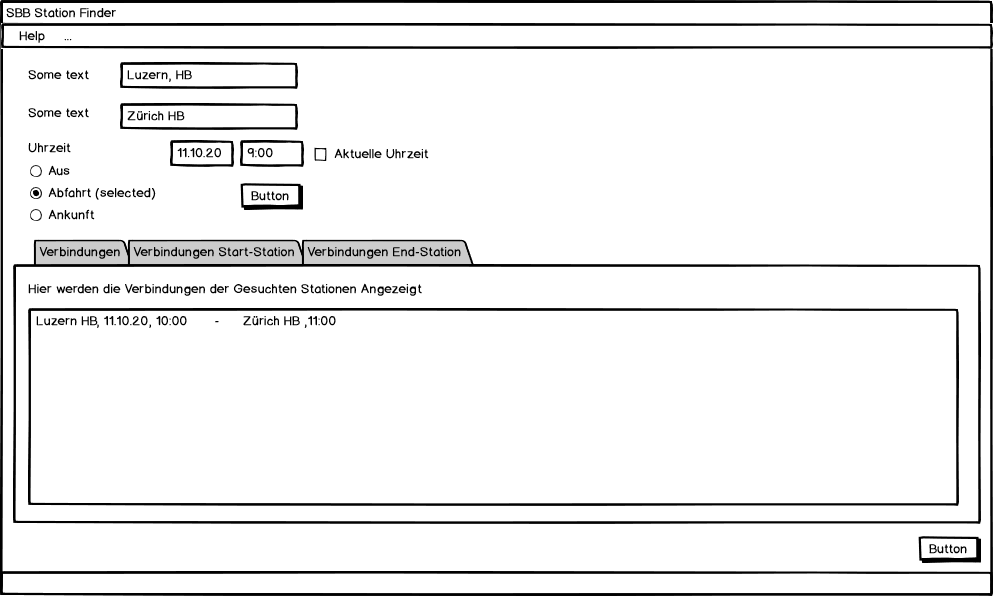
4) SBBStationFinder.exe ausführen

1. Anforderungen
   1. Kundenanforderungen

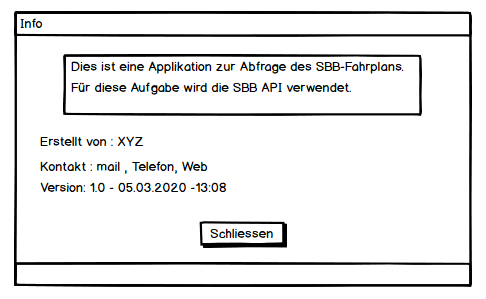


Quelle: ICT-Berufsbildung

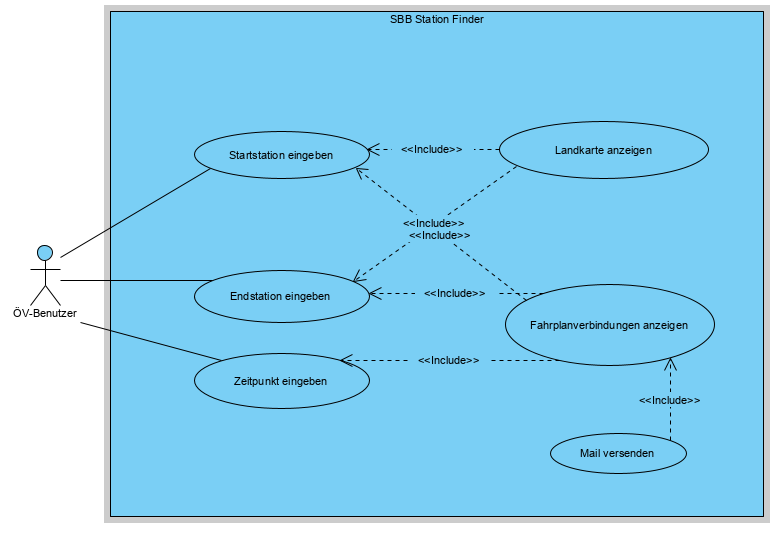
1. Grafische Darstellung
   1. Mokup - frmMain



* 1. Mokup - frmInfo



1. Use Case
   1. Diagramm



* 1. Beschreibung - Stationen eingeben

|  |  |
| --- | --- |
| **Use Case** | **Startstation, Endstation eingeben** |
| **Beschreibung** | Der Benutzer kann über die Eingabefelder Stationen definieren die bei der Suche nach Fahrplanverbindungen verwendet werden. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der Benutzer möchte Informationen zu Station- oder Fahrplan-Verbindungen abfragen. |
| **Vorbedingungen** | keine |
| **Ablauf** | Der Benutzer kann die Stationen wie folgt eingeben:   * Nur eine Startstation * Nur eine Endstation * Start- und End-Station   Wenn einer dieser Punkte erfüllt ist kann nach Verbindungsinformationen gesucht werden. |
| **Ergebnis** | Suche kann erfolgen |

* 1. Beschreibung - Zeitpunkt eingeben

|  |  |
| --- | --- |
| **Use Case** | **Zeitpunkt eingeben** |
| **Beschreibung** | Der Benuter kann über die Eingabefelder ein Datum und eine Uhrzeit definieren die bei der Verbindungssuche verwendet werden. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der Benutzer möchte Station oder Fahrplan-Verbindungen für einen apezifischen Zeitpunkt abfragen. |
| **Vorbedingungen** | Use Case : Startstation, Endstation eingeben |
| **Ablauf** | 1. Der Benutzer muss auswählen ob die Zeitangaben einer Abfahrt oder Ankunft zugeordnet werden sollen. Standardeinstellung ist die Abfahrtszeit |
| **Ergebnis** | Suche kann erfolgen |

* 1. Beschreibung - Fahrplanverbindungen anzeigen

|  |  |
| --- | --- |
| **Use Case** | **Fahrplanverbindungen anzeigen** |
| **Beschreibung** | Der Benutzer kann über die Suche Fahrplaninformationen anfordern und anzeigen lassen. |
| **Akteur(e)** | ÖV-Benutzer |
| **Auslöser** | Der Benutzer möchte nach Stations- oder Verbindungs-Informationen suchen. |
| **Vorbedingungen** | Use Case : Startstation, Endstation eingeben  Use Case : Zeitpunkt eingeben |
| **Ablauf** | 1. Der Suche-Button wird betätigt 2. Die eingegebenen Informationen werden auf korrektheit und notwendigkeit überprüft. 3. Die Abfrage der API-Erfolgt |
| **Hinweis** | * Ist eine Abfahrtstation definiert können nur die Verbindungen der Abfahrtstation angezeigt werden. * Ist eine Ankunftstation definiert können nur die Verbindungen der Ankunftstationen angezeigt werden. * Ist eine Abfahrt- und Ankunft-Station definiert können die Verbindungen beider Stationen angezeigt werden. Dazu können die möglichen Verbindungen der einzelnen Stationen angezeigt werden. |
| **Ergebnis** | Station- oder Fahrplan-Informationen werden angezeigt |

1. Programmierrichtlinien
   1. Benennung der visuellen Elemente

Folgende Bennennungsrichtlinien gelten für visuelle Komponenten.

|  |  |
| --- | --- |
| **Komponente** | **Richtline , Beispiel** |
| Form | frmX , frmMain |
| Button | btnX , btnClose |
| TextEdit | edtX , edtVorname |
| DateTimePicker | dtpX , dtpJahrgang |

* 1. Codierrichtlinien

\*Da es manchmal schwierig ist ein geeignetes Wort für eine Variable zu finden verwende ich Wörter aus dem Deutschen oder Englischen. Dazu möchte ich umlaute im Code vermeiden:

Beispiel: Öffnen -> ~~fileÖffnen~~ -> fileOpen

|  |  |
| --- | --- |
| **Beschreibung** | **Richtline , Beispiel** |
| { } | Werden auf einer neuen Zeile geschreiben |
| Variablen | Beginnen immer mit einem Kleinbuchstaben, weitere Wörter weden mit einem Grossbuchstaben begonnen  Beispiel: personVorname |
| Variablen (Parameter) bei eigenen Funktionen oder Methoden deklaration. (Nicht beim Aufruf!) | Beginnen immer mit einem "**\_**"  Beispiel:  void Funktion2( string \_a)  {  Console.WriteLine("Ich habe {0} mitgegeben", \_a);  } |
| Variablennamen | \*Werden Englisch oder Deutsch geschreiben. |
| Umlaute | Umlaute sind nur in Kommentaren erlaubt |

1. Umgesetzte Anforderungen

|  |  |  |  |
| --- | --- | --- | --- |
| **Priorität** | **ID** | **Beschreibung** | **Erledigt** |
| 1 | A001 | Start- und Endstation mittels Textsuche suchen | OK |
| 1 | A002 | Verbindungen zwischen Stationen anzeigen. | OK |
| 1 | A003 | Abfahrtstafeln anzeigen | OK |
| 2 | A004 | Autocomplete für Suchfelder | OK |
| 2 | A005 | Verbindungen mit Zeitpunkt suchen | OK |
| 3 | A006 | Station auf Karte anzeigen | OK |
| 3 | A007 | Stationen in der Nähe finden | NEIN |
| 3 | A008 | Resultate via Mail versenden | OK |

* 1. Bekannte Bugs
* A004

Die Autocomplete funktion ist nicht perfekt. Bei sehr schneller Eingabe wird der bereits eingegebene Text gelöscht. Dies kann nur behoben werden wenn die ComboBox spezielle angepasst wird.

1. Manueller Test
   1. Use Case 1 - Startstation, Endstation eingeben

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Start Station : Luzern | 1. In die ComboBox(Start) den Text "Luzern" eingeben. 2. Die ComboBox(Ziel) muss leer sein! 3. Suche Button mit Maus betätigen. | Verbindungen ab Luzern werden angezeigt |

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Ziel Station : Zürich | 1. In die ComboBox(Ziel) den Text "Zürich" eingeben. 2. Die ComboBox(Start) muss leer sein! 3. Suche Button mit Maus betätigen. | Verbindungen ab Zürich werden angezeigt |

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Start Station : Luzern  Ziel Station : Zürich | 1. In die ComboBox(Start) den Text "Luzern" eingeben. 2. In die ComboBox(Ziel) den Text "Zürich" eingeben. 3. Suche Button mit Maus betätigen. | Verbindungen zwischen Luzern und Zürich werden angezeigt. Dazu werden alle Verbindungen ab Zürich und ab Luzern angezeigt in den entsprechenden Tabs. |

* 1. Use Case 2 - Zeitpunkt eingeben

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Start Station : Luzern  Ziel Station : Zürich  Datum : 01.01.2020  Zeit: 18:00 | 1. In die ComboBox(Start) den Text "Luzern" eingeben. 2. In die ComboBox(Ziel) den Text "Zürich" eingeben. 3. Den RadioButton(Abfahrt) Einschalten 4. In den DateTimePicker(Datum) "01.01.2020" eintragen. 5. In den DateTimePicker(Uhrzeit) "18:00" eintragen. 6. Suche Button mit Maus betätigen. | Verbindungen zwischen Luzern und Zürich werden angezeigt ab dem entsprechenden Zeitpunkt. Kontrollieren ob die Abfahrtszeit plausibel ist. |

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Start Station : Luzern  Ziel Station : Zürich  Datum : 01.01.2020  Zeit: 18:00 | 1. In die ComboBox(Start) den Text "Luzern" eingeben. 2. In die ComboBox(Ziel) den Text "Zürich" eingeben. 3. Den RadioButton(Ankunft) Einschalten 4. In den DateTimePicker(Datum) "01.01.2020" eintragen. 5. In den DateTimePicker(Uhrzeit) "18:00" eintragen. 6. Suche Button mit Maus betätigen. | Verbindungen zwischen Luzern und Zürich werden angezeigt ab dem entsprechenden Zeitpunkt. Kontrollieren ob die Ankunftszeit plausibel ist. |

* 1. Use Case 3 - Fahrplanverbindungen anzeigen

|  |  |  |
| --- | --- | --- |
| **Testdaten** | **Testablauf** | **Resultat** |
| Start Station : Luzern  Ziel Station : Zürich  Datum : 01.01.2020  Zeit: 18:00 | Wenn die Manuellen Tests für UseCase 1 & 2 erfogreich sind, sind alle Punkte für Use Case 3 auch erfüllt | OK |

* 1. Erweiterte Tests

|  |  |
| --- | --- |
| **Test** | **Resultat** |
| Programm ohne Internetverbindung getestet ?   * Abfrage * E-Mail * Map | OK |
| Ungültige Eingaben Getestet ?   * Hauptformular * E-Mail | OK |
| Tabulatoren reihenfolge Getestet ? | OK |
| Programm ist ohne Maus bedienbar ? | OK |
| Versenden von E-Mail möglich ?   * getestet mit Bluewin account | OK |
| Aufruf der Map getestet ? | OK |
| Visual Studio API Tests | OK |

1. Kürzübersicht über eigene Methoden und Funktionen
   1. frmMain.cs

private void autoComplete()

Methode für die Combobox-Suche damit die Vorschläge gemacht werden.

private Stations getValidSations(string \_s)

Mit dieser Funktion werden nur Stationen zurückgegeben die eine ID und Namen besitzen.

private void dateAndTimeGUIControl()

Methode zum Ein- und Aus-schalten der Zeit-Gui-Controls.

private void getDateTimeConnection(string \_start, string \_end, string \_date, string \_time, short \_isArrival)

Methode zum Anzeigen und abfragen der Verbindungen wenn ein Zeitpunkt angegeben wurde.

private void getStationConnection(string \_start, string \_end)

Methode zum Anzeigen und abfragen der Verbindungen wenn kein Zeitpunkt angegeben wurde.

private StationBoardRoot getStationBoard(ComboBox \_cbSender, ListBox \_lb, TabPage \_tb)

Funktion zum Anzeigen und abfragen der Stationsinformationenen.

private string convertToAPIDate(string \_string)

Funktion für die Datumskonvertierung in das entsprechende API-Format.

private void openBrowser(string \_x, string \_y)

Methode um den Standardbrowser. Dieser Zeigt den Standort an den mitgegebenen x,y Koordinaten.

* 1. frmMail.cs

public frmMail(ListBox \_Connection, ListBox \_sbStart, ListBox \_sbZiel)

Erweiterter Konstruktor um die Informationen übergeben zu können.

private void fillMailContent(ListBox \_lb)

Methode zum befüllen der TexBox die die MailInformationen beinhaltet.

1. API Erweiterung

Um Verbindungen an einem spezifischen Zeitpunkt zu suchen war es notwendig die API anzupassen. Es wurden keine bestehenden API Funktionen oder Methoden geändert. Die API wurde nur erweitert.

Neu:

Transport.cs -> public Connection GetConnectionByDateTime(…)

TransprtTest.cs -> public void ConnectionsByDateTime()

Itransport.cs -> Connections GetConnectionByDateTime(…)