Advanced travel

M318 Objektbasiert Programmieren

ÜK 318

Marco Frautschi

# Einleitung:

Wir haben im ÜK 318 den Auftrag bekommen, eine Fahrplan-Desktopapplikation zu erstellen.

Diese Dokumentation dokumentiert die Funktion der und der Anwendungsgebrauch, der Applikation wird getestet.

Folgende Anforderungen wurden an die Applikation gestellt:

|  |  |
| --- | --- |
| ID | Beschreibung |
| A001 | 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. |
| A004 | Als ÖV-Benutzer möchte ich, dass schon während meiner Eingabe erste Suchresultate erscheinen, damit ich effizienter nach Stationen suchen 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. |

A001 – A003 sind Pflicht.

A004 – A005 sind zu empfehlen.

A006 – A008 sind Zusatzaufgaben.

# Anforderungen umgesetzt:

Ich habe folgende Anforderungen umgesetzt:

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

# Fehlende Funktionen und Bugs:

To Many Requests:

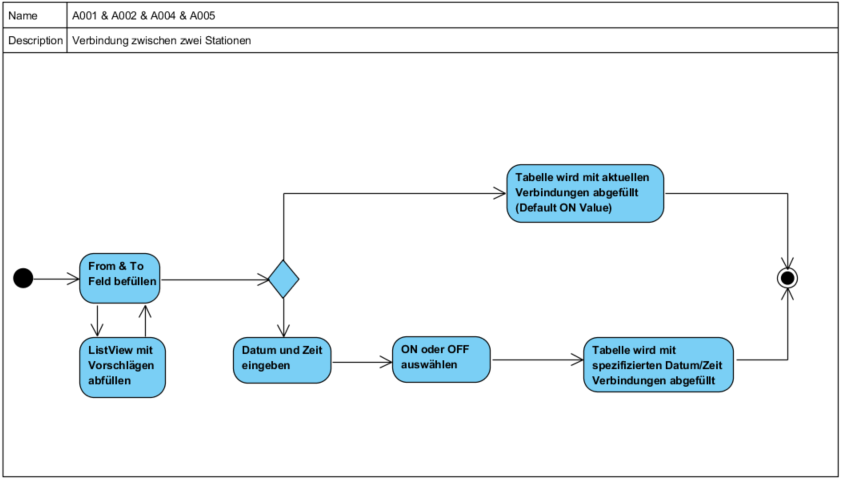
Ich habe versucht diese Exception zu verhindern, indem ich in meinem eigenerstellten Controll (MySearchField) die Zeit speichere, in der meine letzte Anfrage an den Server gesendet wurde. Jedoch bin ich nur auf Sekundengenauigkeit gekommen, daher kann man nur eine Request pro Sekunde senden. Dies funktioniert, wenn man den Text im Feld korrigieren möchte (Backspace), leider nicht besonders gut. Deshalb habe ich, damit es den Benutzer nicht irritiert, in der API eine try{}catch{} Klausel gesetzt, und die WebException aufgefangen. In diesem Catch fange ich den Fehler ab, jedoch wird er nicht behandelt und nicht weitergegeben.

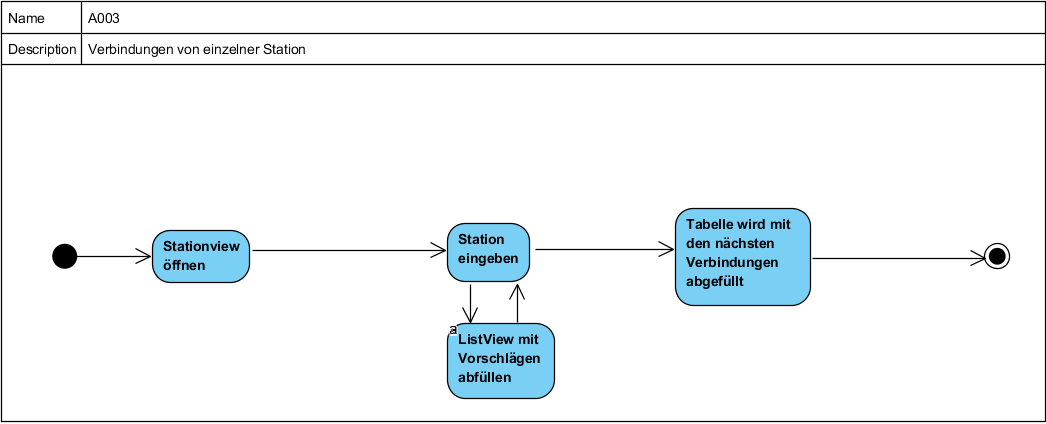
# Diagramme:

## Use Case:

## 

## Aktivitätsdiagramm:





# Testfälle:

Bei folgenden Testfällen wird davon ausgegangen, dass Sie das Programm neu gestartet haben.

## Testfall 1:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Beim From-Feld «Luz» eingeben. | ListBox mit Vorschlägen zu Haltestellen erscheint |
| 2 | (Nach Schritt 1) Luzern mit Maus auswählen | ListBox schliesst sich und ausgewählter Wert ist in From-Feld. |
| 3 | Beim To-Feld «Rothenburg Dorf» eingeben und mit der Tastatur «Rothenburg, Dorf» auswählen | ListBox schliesst sich und ausgewählter Wert ist in To-Feld. |
| 4 | (Nach Schritt 3) Button-Now klicken | Im DataGridView erscheinen die aktuellen Verbindungen der zwei Stationen. |
| 5 | Taste Alt + E Drücken | Applikation schliesst sich. |

## Testfall 2:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | From-Feld mit «Luzern» befüllen, To-Feld mit «Rothenburg, Dorf» befüllen und Datum Dropdown öffnen und 11.Oktober.2017 auswählen | Datum 11.Oktober.2017 steht im Datum-Feld |
| 2 | Im Time Feld 10:50 auswählen (mit Tastatur oder Maus) | Gesetzte Zeit ist im Zeit-Feld. |
| 3 | On Button auswählen | Farbe des ausgewählten Buttons wird blau. Der andere Button ist grau. |
| 4 | Search-Button Klicken | DataGridView wird mit 4 Datensätzen der entsprechenden Verbindung abgefüllt. |
| 5 | Taste Alt + C Drücken | Applikation schliesst sich |

## Testfall 3:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Button «Search for Station» drücken | Stationsansicht öffnet sich. |
| 2 | Bei Station «Luz» eingeben und bei der ListBox Luzern auswählen | Im Station-Feld ist der ausgewählte Wert. |
| 3 | Button «Search» klicken | DataGridView wird mit den aktuellen Stationsverbindungen abgefüllt. |

## Testfall 4:

|  |  |  |
| --- | --- | --- |
| Schritt | Aktivität | Erwartetes Resultat |
| 1 | Button-DE klicken | Alle Labels und Buttons ändern ihre Sprache auf Deutsch |
| 2 | Button-FR klicken | Alle Labels und Buttons ändern ihre Sprache auf Französisch |
| 3 | Button-EN klicken | Alle Labels und Buttons ändern ihre Sprache auf Englisch |

# Installationsanleitung:

# SelfmadeFunctions:

Mehrsprachig:

Ich habe als eigenes «Feature», die Mehrsprachigkeit meines Programms programmiert. Oben rechts hat es 3 Buttons (Button-DE, Button-EN und Button-FR), bei denen man die Sprache auf Deutsch, Englisch oder Französisch setzen kann. Standard ist Englisch.

Leider konnte ich die DataGridwView-Headernamen nicht verändern, dies folgt in deinem späteren Update.

# Programmierrichtlinien:

## Namenskonventionen:

Alle Variablen fangen mit einem kleinen Buchstaben an und mit [Camel Case](https://en.wikipedia.org/wiki/Camel_case) weitergeführt.

Membervariablen fangen mit einem kleinen m an.



### GUI-Controls:

Buttons beginnen mit: btn\* 🡪 btnSave

TextFields beginne mit: txt\* 🡪 txtDestination

Label beginnen mit: lbl\* 🡪 lblSave

DateTimePicker beginnen mit: dtp\* 🡪 dtpDate

DataGridView beginnen mit: dataGridView\* 🡪 dataGridViewMain

Methoden werden mit einem grossen Buchstaben angefangen.

GUI-Controller-Actionmethoden werden mit dem jeweiligen Codekürzel angefangen. (siehe Oben)

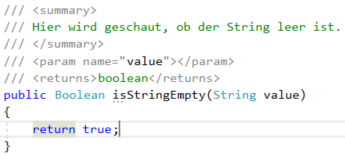
## Deklaration:

Alle Variablen werden unter der Funktionsdefinition definiert.

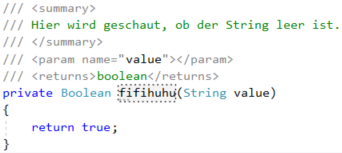
Variablen müssen nicht von Anfang an initialisiert werden.

## Kommentare:

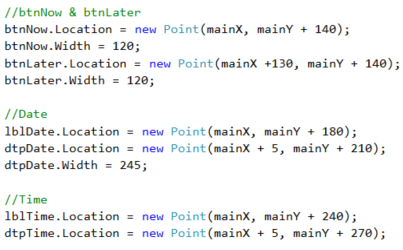
Alle public Methoden werden kommentiert mit 3 x ///.



Falls der Name der Methode nicht selbsterklärend ist, wird die Methode mit 3 x /// kommentiert.



Zur Übersichtlichkeit der verschiedenen Controls, Kommentiere ich die Namen der verschiedenen Controls ein.



## Statement (New Lines, Indentation):

Alle Geschweiften Klammern «{ }» werden auf eine neue Linie geschrieben.

Nach einer Geschweiften Klammer «{» wird eingereiht.