ÖV-Applikation

ÜK-318

Levin Joller

27. November 2018

Inhalt

[Einleitung 3](#_Toc531173367)

[Zweck der Dokumentation 4](#_Toc531173368)

[Mockup 5](#_Toc531173369)

[Umsetzung 6](#_Toc531173370)

[Programmierrichtlinien 6](#_Toc531173371)

[Allgemein 6](#_Toc531173372)

[Properties 6](#_Toc531173373)

[Methoden 6](#_Toc531173374)

[Klassen 6](#_Toc531173375)

[Kommentare 7](#_Toc531173376)

[GUI-Controls 7](#_Toc531173377)

[Quellen 8](#_Toc531173378)

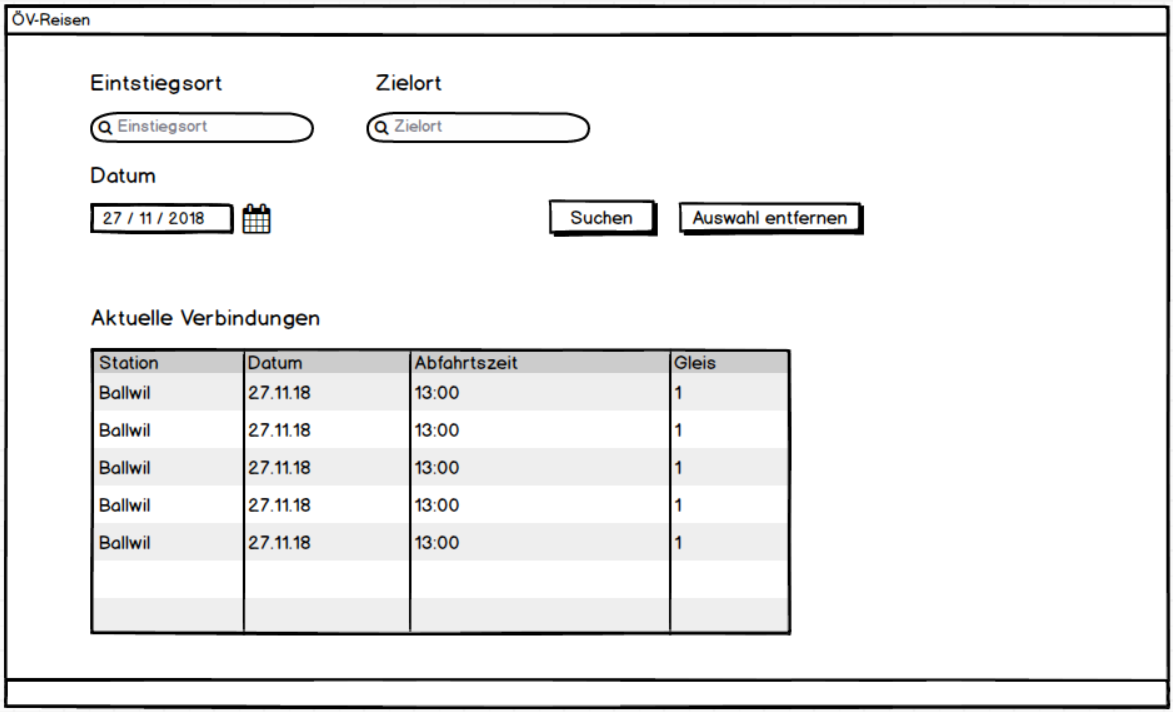
# Einleitung

Ziel der Arbeit ist es, eine Applikation zur Anzeige von Informationen über den aktuellen Bankverkehr nach Vorgaben zu schreiben. Dafür verwende ich die Programmiersprache C# und die dazu passende Entwicklungsumgebung namens Visual Studio von Microsoft.

# Zweck der Dokumentation

Die Dokumentation soll dazu dienen nachzuvollziehen, wie meine Applikation aufgebaut ist und was meine Überlegungen dazu waren.

# Mockup



Dies sind meine Vorstellungen vor der Umsetzung des Projekts, damit ich einen Anhaltspunkt habe wie ich es in etwa umsetzten muss. Ich habe versucht die Grafische Benutzeroberfläche so schlicht wie möglich zu gestalten, aber immer noch selbsterklärend. Der Benutzer sollte seinen Einstiegsort, also seinen Abfahrtsort eintragen können und alle Züge

# Umsetzung

# Programmierrichtlinien

Der Zweck der Programmierrichtlinien ist es, dass der Programmcode Einheitlich ist und somit Änderungen auch von anderen Personen einfach vorzunehmen sind. Damit ein Caos im Code vermieden wird und die Arbeit im Team vereinfacht braucht man solche Richtlinien. Zudem sollen auch Personen, die den Code selbst nicht geschrieben haben den Code mit wenig Aufwand verstehen können. Damit wir zusätzlicher Zeitaufwand zum Verstehen des Codes vermieden.

Folgende Punkte sollten beim Code schreiben beachtet werden:

Lesbarkeit, Wartbarkeit, Übersichtlichkeit, Verständlichkeit

## Allgemein

Es werden Tabs für Einrückungen verwendet anstatt Spaces (Leertaste).

Es wird alles in Englisch geschrieben, ausser die Kommentare -> Deutsch Naming Conventions Variablen

Variablen verwenden camelCase und werden gleich beim Deklarieren initialisiert. Variablen verwenden keine Abkürzungen. (usrGrp -> userGroup)

Beispiel:

string namingConventions = null;

falsches Beispiel:

string namingConventions;

## Properties

Properties sind in PascalCase geschrieben und sind plural.

public string FirstName { get; set;}

## Methoden

Methoden werden in PascalCase geschrieben. Methoden fangen mit einem Verb an. Brackets fangen auf einer neuen Linie an.

Beispiel:

DoSomething () { //code }

## Klassen

Klassen fangen mit einem Grossbuchstaben an und sind Singular.

public class Tree { //Properties }

## Kommentare

Kurze und verständliche Kommentare vor Methoden, WENN Methodename nicht selbsterklärend ist. -> Deutsch. Bei allen Public Methoden muss ///<summary> als Kommentar stehen. Bei Returns muss ///<returns> Rückgabewert </retruns> stehen.

Beispiel:

// Gibt den string in UPPERCASE zurück. DoSomething (string text) { // code }

## GUI-Controls

Folgenden Controls starten mit folgenden Prefix und danach mit einem Grossbuchstaben.

Label: lbl... Button: btn... ComboBox: cbo... TextBox: txt... ListBox: lst... PictureBox: pic... Timer: tmr... RadioButton: opt... CheckBox: chk... GroupBox: grp... Panel: pnl...

GUI-Events: Standard.

Beispiel:

lblName btnBerechnen

# Quellen