Tour Planner – Aida Omic

Design

Grundsätzlich wurden alle Anforderungen erfüllt, bis auf die Searchbar. Diese ist case sensitive implementiert. Außerdem wurde die documentation-annotation aus Zeitgründen nicht implementiert.

Als file-type für den Import und Export von Daten wurde PDF gewählt. Beim Import File handelt es sich um ein einfaches Pdf-File in dem die Werte durch ; getrennt sind. Beim Export-File bleibt dem Nutzer überlassen, ob er die Daten in dem Format des Import-Files extrahiert haben möchte oder im Format eines Tables.

Um die Pdf-Dateien zu generieren wurde sowohl beim Export der Daten als auch beim Generieren des Reports IText7 verwendet.

Als Zusatz-Features ist ein 'Hilfe'-Fenster implementiert und eine Editier-Möglichkeit der Tour Logs geboten. Als reusable UI-Component wurde die View für die Darstellung der Maps gewählt.

Architektur

Grundsätzlich wurde das MVVM-Pattern verwendet. Somit bildet sich folgender Aufbau:

- View
- Controller
- View Model
- Business Layer
- Data Access Layer

Diese Layer sind auch als Packages in der Projektstruktur abgebildet. Unter dem TourPlanner-Package sind die Klassen zum Main-Window und die Log bzw Tour Klasse zu finden.

Alle Packages zum MVVM-Pattern sind jeweils nochmals unterteilt. Beispielsweise sind die Views unterteilt in ‚Tour', ‚Tour Log', ‚Logging' und ‚Sonstige', um die Projektstruktur somit übersichtlicher zu gestalten. Auch der Business Layer wurde in einzelne Unterordner unterteilt. Hierbei wurde versucht die Klassen nach ihren Aufgabengebieten zu unterteilen.

Design Patterns

Die gewählten Design-Patterns sind Delegation und Factory. Das Factory-Pattern war vor allem sehr hilfreich bei der Implementierung des Data Access Layers. Auch bei den Klassen, die zum Importieren bzw. Exportieren der Daten verwendet werden, wurde das Factory-Pattern angewendet.

Das Delegation-Pattern wurde fast durchgehend im Business Layer verwendet. Die meisten Klassen delegieren ihre Aufgaben an andere Hilfsklassen.

Failures

Der einzige Punkt, der bis zum Schluss unerfüllt bleibt, ist das ändern der Logging Levels. Logging funktioniert zwar einwandfrei, jedoch kann das Logging Level nicht geändert werden, da es einen Fehler in der CodeLogik gibt, den ich nicht lösen konnte.

Unit Tests

Für die Unit Tests wurden Klassen des Business und Data Access Layers gewählt. Vom Data Access Layer werden sowohl die Klassen für die Tours als auch für die Tour Logs getestet. Hierbei wird bei beiden getestet, ob eine Tour bzw. ein Tour Log erstellt wird, die Daten richtig aus der Datenbank gelesen werden und ob die Tour bzw. der Tour Log wieder richtig gelöscht wird. Ein Testfall der nur bei den Tours vorkommt ist das Editieren. Hierbei wird überprüft ob die Datenbank richtig geupdatet wird, je nachdem welche Daten geändert wurden.

Im Business Layer werden verschiedene Klassen getestet: AverageCalculator, CounterHandler, PathHandler, PropertyHandler und MapQuestTest.

In der Testklasse AverageCalculatorTest werden alle Methoden getestet, die die Durchschnittswerte für den Report berechnen. Diese Klasse wurde gewählt, da beim Implementieren des Exportieren eines Reports ein paar Ungereimtheiten in den Ausgabedaten vorgekommen sind und ich mir so leichter getan habe, den Fehler zu finden, als jedes Mal die Applikation zu starten. Die Testfälle der Klasse CounterHandlerTest wurden im Zuge der Implementation der Testklasse AverageCalculatorTest geschrieben.

Die Testfälle der Testklasse PathHandlerTest wurden implementiert während ich Refactoring durchgeführt habe, um einerseits die PathHandler-Klasse etwas zu optimieren und andererseits die Fenstergrößen beim Laden einer neuen View richtig zu stellen.

Die PropertyHandlerTest-Klasse testet, ob einerseits die SQL-Statements und andererseits die MapQuest-Url richtig aus dem config-file gelesen werden.

Zu guter Letzt testet die MapQuestTest-Kasse die verbindung zu MapQuest, ob diese die Daten richtig retrieved. Diese Klasse wurde eigentlich nicht implementiert, um die bestehende Verbindung zu testen, da ich ja bereits wusste das das problemlos funktioniert und es dabei auch nie Probleme gab. Die Test-Klasse wurde implementiert um bestehende Testfälle zu haben, die abgeändert werden können, falls in späterer Folge beispielsweise etwas an der URL geändert wird (durch MapQuest selbst oder durch Hinzufügen von weiteren Attributen), und sich somit die Arbeit in Zukunft zu erleichtern.

Tracked Time

- 03/03/2021
    - Initial Commit und SetUp des Remote-Repo (1h)
- 05/03/2021
    - MVVM angeschaut und ‚Hello world' implementiert (1h)
- 07/03/2021
    - UI Grundgerüst (MainWindow + Controller) (3h)
    - Buttons ändern ein Textfeld (1h)
- 01/04/2021
    - UI & DataBinding überarbeitet (2h)
- 14/04/2021
    - Andere Views laden (2h)
- 15/04/2021
    - Hinzufügen einer Toru begonnen (2h)
    - Map Quest implementiert und funktioniert (3h)
- 18/04/2021
    - Datenbank und -verbindung erstellt (1h)
    - Tour Daten ein und auslesen funktioniert (2h)
- 20/04/2021
    - UnitTests begonnen (20min)
    - Hinzufügen und editieren von Tours funktioniert (2h)
    - Merging branches (1h)
- 21/04/2021
    - Delete Tour funktioniert (1h)
    - SeacrhBar für Tours funktioniert (1h)
    - Image Window hinzugefügt (1h)
- 22/04/2021
    - Distance zu Tour (in Datenbank und Projekt) hinzugefügt (2h)
- 27/04/2021
    - Config-file hnzugefügt + Daten auslesen (1h)
    - SQL-Query zu config-file hinzugefügt (1h)
    - MapQuest-URL zu config-file hinzugefügt (30 min)
- 30/04/2021
    - Itext7 angeschaut und hinzugefügt (2h)
    - Tour export funktioniert (2h)
    - Tour import angefangen (2h)
- 03/05/2021
    - Tour import from pdf funktioniert(4h)
- 12/05/2021
    - Refactoring Business Layer (2h)
    - Copy Tour (1h)
- 17/05/2021
    - ViewModels aufgeteilt + refactoring(1h)
    - UIs für TourLogs erstellt (2h)
- 18/05/2021
    - Adding TourLog (3h)

- 19/05/2021
    - Tour Logs import/export (3h)
    - SearchBar (1h)
    - Tour Log edit und delete (2h)
- 27/05/2021
    - UnitTests für DAL (1h)
- 01/06/2021
    - Logging begonnen (1h)
    - Logs hinzugefügt (1h)
- 02/06/2021
    - Change Logging Level versucht (3h)
    - Report generieren für eine oder alle Tours (3h)
- 03/06/2021
    - HelpWindow erstellt (alles) (2h)
    - Refactoring Tours und TourLogs (2h)
    - Refactoring (Aufteilen der Klassen) (2h)
    - Überarbeiten der Tests (30 min)
- 04/06/2021
    - Refactoring Projekt (1h)
    - Überarbeiten der Tests (30 min)