Shape

Description automatically generated

PROJEKTDOKUMENTATION

zu

**myTracks**

Version 1.0 vom 31.07.2022

Entwickelt von Huber Andreas, Schwarz Mathias und Schwarz Markus

Unter Aufsicht von a. Univ.-Prof. Dr. Johannes Sametinger

GitHub: <https://github.com/Trooooooonix/SE-Internship>



**Inhaltsverzeichnis**

[1 Ziele des Projektes 3](#_Toc110087489)

[2 SOLL - IST Vergleich 3](#_Toc110087490)

[2.1 SOLL - Zustand 3](#_Toc110087491)

[2.2 IST – Zustand 4](#_Toc110087492)

[3 Probleme bei der Ausarbeitung 5](#_Toc110087493)

[3.1 Projekt in JAR-Archiv überführen 5](#_Toc110087494)

[3.2 Exceptions bei implementierung des Parsers 5](#_Toc110087495)

[3.3 Testen 5](#_Toc110087496)

[4 Projektverlauf 6](#_Toc110087497)

[4.1 Erster Sprint 6](#_Toc110087498)

[4.2 Zweiter Sprint 7](#_Toc110087499)

[4.3 Dritter Sprint 7](#_Toc110087500)

[5 Kommunikation und Aufzeichnung 7](#_Toc110087501)

[5.1 Auslesen von GPX Dateien 7](#_Toc110087502)

[5.2 Anzeige von anderen Charts (Beispielsweise LineChart) 7](#_Toc110087503)

[5.3 JavaFX und Maven update 7](#_Toc110087504)

[5.4 Google Maps API integration 7](#_Toc110087505)

[5.5 Darstellung aggregierter Daten 8](#_Toc110087506)

[6 Fazit 8](#_Toc110087507)

[7 Abbildungsverzeichnis 9](#_Toc110087508)

1. Ziele des Projektes

Die Ziele dieses Projektes waren ein voll funktionsfähiges Programm inklusive Benutzerhandbuch und Dokumentationen zu erstellen, welches TCX, sowie GPX Dateien auslesen und anzeigen kann. Weiters sollten Funktionen implementiert werden, die die Handhabung der eingelesenen Daten erleichtert bzw die Übersicht verbessert.

1. SOLL - IST Vergleich

In den folgenden Punkten wird näher auf die Funktionen, die das Programm mit sich bringen soll, eingegangen, dabei wird differenziert zwischen SOLL und IST.

* 1. SOLL - Zustand
     1. Directory mit Dateien auswählen

Das Programm soll ein Directory auswählen können, woraus die TCX und GPX Dateien ausgelesen werden können, die dann in dem User Interface angezeigt werden.

* + 1. Auslesen und Anzeigen von TCX und GPX Dateien

Wie im Punkt 2.1.1. erklärt, soll das Programm TCX und GPX Dateien mit allen relevanten Informationen auslesen und im User Interface anzeigen können.

* + 1. Daten auf Monat und Jahr aggregieren

Anhand einer groupingBy-Function soll das Programm die Daten auf Monat und Jahr aggregieren und anzeigen können.

* + 1. Barchart mit (aggregierten) Daten anzeigen

Im User Interface des Programms soll, um zusätzlich einen visuellen Reitz zu schaffen, ein Barchart angezeigt werden, welches sowohl aggregierte als auch detailierte Daten über die einzelnen Tracks anzeigen soll.

* + 1. Filtern nach Sportart

Da die Tracks in mehreren Sportarten angesiedelt sein können, soll auch eine filterBy-Funktion eingebaut werden, welche nur gewünschte Sportarten anzeigt. Per Default sollen hier alle Sportarten angezeigt werden.

* + 1. Spalten ausblenden

Um die User-Experience zu verbessern, soll der Anwender Spalten ausblenden können um möglicherweise redundante Daten (für den User) ausblenden zu können.

* 1. IST – Zustand
     1. Directory mit Dateien auswählen

Zusätzlich zu der Funktion des auswählen der Directories wurde im Programm eine Root-Directory festgelegt, die beim initialen Start des Programms einen Ordner inklusive ReadMe- und Demo- Datei erstellt. Dieses Root-Directory kann auf Belieben vom Anwender verändert werden.

* + 1. Auslesen und Anzeigen von TCX und GPX Dateien

Da in den Beispiel-Dateien hauptsächlich TCX Dateien enthalten waren, entschieden wir uns vorerst nur einen TCX-Parser umzusetzen. Die Auslese der GPX Dateien ist, auf Grund von mangelnden Beispieldateien, für zukünftige Releases angedacht.

* + 1. Daten auf Monat und Jahr aggregieren

Diese Funktion, wie im Punkt 2.1.3. beschrieben, wurde vollends umgesetzt. Die eingelesenen Daten können auf Monat und Jahr aggregiert werden, wobei bei der Aggregation des Monats zusätzlich nach Jahren differenziert wird (Juli [2020] und Juli [2021]).

* + 1. Barchart mit (aggregierten) Daten anzeigen

Die Anzeige des Barcharts wurde eingeführt. Hierbei werden die Daten von den einzelnen Tracks in einem Barchart angezeigt wobei zusätzlich für weitere Releases angedacht ist, dass auch eine Jahresübersicht mit den aggregierten Daten dargestellt werden kann.

* + 1. Filtern nach Sportart

Wie im Punkt 2.1.1.5. erläutert, wurde diese Funktion vollends umgesetzt. Als Default werden alle Tracks, nach neuestem Track (Datum) sortiert, angezeigt.

* + 1. Spalten ausblenden

Diese Funktion wurde, wie im Punkt 2.1.1.6. erläutert, im Programm implementiert.

1. Probleme bei der Ausarbeitung

Wie in jedem Projekt kam es auch bei diesem Projekt zu den ein oder anderen Schwierigkeiten bei der Ausarbeitung. In diesem Punkt wird auf ein paar der aufgetretenen Probleme eingegangen.

* 1. Projekt in JAR-Archiv überführen

Genauer wurde beim erstellen des Archivs die Struktur innerhalb des Projektes verworfen wodurch Probleme bei der Datenauslese sowie finden von Images, die wir für das Programm eingefügt hatten, auftraten. Dies resultierte in FileNotFound-Exceptions welche das Programm zum Absturz brachten. Hierfür wurden die Methoden „readProperties()“ und „getPropertiesAsString()“ implementiert.

* 1. Exceptions bei implementierung des Parsers

Beim erstellen des Parsers hatten wir anfänglich Probleme bei der Datenauslese, da die Variablen bzw. verschiedenen Stellen innerhalb der TCX-Dateien gleich benannt waren, welches zu einem verfälschtem Ergebnis führte. Beispielsweise „DistanceMeters“ war sowohl in den einzelnen Laps als auch in den Trackpoints enthalten. Weiters hatten wir Probleme bei der Auslese von dem Datum des Tracks, da hier verschiedene Formate verwendet wurden, welches wieder in einer geworfenen Exception resultierte.

* 1. Testen

Bei den JUnit Tests viel es uns sehr schwer gute Testfälle zu finden. Hauptgrund ist, dass wir hauptsächlich bereits vorhandene Java Bibliotheken verwendet haben und diese von den Entwicklern schon getestet sind. Wir haben nur ein paar Methoden für Berechnungen selbst geschrieben.

1. Projektverlauf

Graphical user interface

Description automatically generated with low confidenceWie in Abbildung 1 ersichtlich war der Projektverlauf der Struktur der LVA angepasst. Dabei wurde das Projekt in drei Sprints aufgeteilt. Diese werden jeweils in den Punkten 4.1, 4.2 und 4.3 genauer erläutert.

Abbildung 1: Meilensteinplan

Graphical user interface, application

Description automatically generatedVor dem ersten Sprint haben wir festgelegt, dass wir Intellij als Entwicklungsumgebung verwenden, sowie Clockify als Zeitaufzeichnungstool. Weiters haben wir ein Grundlegendes Konstrukt des UI’s mit Hilfe des Tools Miro verwendet (siehe Abbildung 2).

Abbildung 2: erster Prototyp

* 1. Erster Sprint

Im ersten Sprint wurde der Fokus eher auf die Grundsätzlichen Dinge des Programmes gelegt. Diese Umfassen vorrangig GUI, Parser sowie die Anzeigen von Tabellen und dem Barchart. Weiters wurden die ersten Funktionen implementiert die beispielsweise Daten aufsummiert, um die Korrektheit der Anzeige zu gewährleisten.

* 1. Zweiter Sprint

Im zweiten Sprint wurden die Ergebnisse des erstens Sprints teilweise überarbeitet und erweitert, Änderungen in der Struktur des Programmes vorgenommen und die Implementierung der „grouping“- und „filter“-Funktionen durchgeführt. Darüber hinaus wurde, aus effizienzgründen, von einem DOM-Parser auf den SAX-Parser umgestellt.

* 1. Dritter Sprint

Im dritten Sprint wurden die Ergebnisse der bisherigen Arbeit finalisiert und überarbeitet. Hierbei lag der Fokus eher auf dem Feinschliff wie auf die Implementierung neuer Funktionen. Weiters wurden in diesem Teil des Praktikums die JavaDoc, Projektdoku und das Benutzerhandbuch geschrieben.

1. Kommunikation und Aufzeichnung

Die Projektinterne Kommunikation verlief grundsätzlich reibungslos und fand über Text- und Voicechannels (Discord oder Whatsapp) statt. Meist wurden gemeinsam über einen Textchannel Termine gefunden zudem alle drei Teammitglieder über einen Voicechannel verbunden an Problemen, Bugs bzw. Funktionen im Programm arbeiteten. Probleme traten dabei eher selten auf.

Die Zeitaufzeichnung wurde ausschließlich in der Webapplikation „Clockify“ (<https://app.clockify.me/tracker>) durchgeführt.

Die Synchronisation des Codes im Team wurde in einem Online-Repository auf der Plattform Github durchgeführt. Dabei konnte jeder der Teammitglieder in seiner eigenen Branche seine Codeveränderungen hochladen und am Ende jeder „Session“ diese in die Master-Branche submitten. Dies konnte dann jedes andere Teammitglied durch Aktualisieren runterladen und an der neueren Version weiterarbeiten.

In den folgenden Unterkapiteln werden die Funktionen beschrieben, welche für folgende Releases geplant sind.

* 1. Auslesen von GPX Dateien

Wie im Punkt 2.1.2.2. erläutert, wird das Auslesen von GPX-Dateien für zukünftige Releases angedacht. Diese soll beim nächsten Release Priorität erhalten, um die Einschränkung auf TCX-Dateien aufzuheben und somit die Funktionalität sowie Attraktivität des Programms erweitern.

* 1. Anzeige von anderen Charts (Beispielsweise LineChart)

Um auch den visuellen Reiz des Programmes zu verbessern, soll es zukünftig möglich sein zwischen verschiedenen Chartoptionen variieren zu können. Beispielsweise soll die „pace“ per Default als LineChart angezeigt werden.

* 1. JavaFX und Maven update

Für ein moderneres User Interface ist geplant auf JavaFX umzustellen und zum besseren handling von Abhängigkeiten wird auf Maven umgestellt. In der Readme.md in GitHub sind dazu mehr Details wie ein Link zu einem Repository dem nicht fertigen JavaFX Projekt.

* 1. Google Maps API integration

Dient, um Activities auf einer Landkarte visualisieren zu können.

* 1. Darstellung aggregierter Daten

Zukünftig soll auch die Funktion der Darstellung aggregierter Daten in einem Barchart implementiert werden. Dabei sollen die Daten, die zuvor auf Monat oder Jahr zusammengefasst wurden, in einem Barchart dargestellt werden können.

1. Fazit

Rückblickend wurden beinahe alle Funktionen, die geplant wurden, auch vollends implementiert. Die Funktionen, die bei dieser Veröffentlichung nicht implementiert wurden, werden, wie im Punkt 6.2. erläutert für zukünftige Releases eingeplant und, je nach Feedback der Anwender, priorisiert umgesetzt. Die Zusammenarbeit der Teammitglieder funktionierte gut und ohne Zwischenfälle. Da weder der Zeitrahmen gesprengt noch Funktionen vernachlässigt wurden, kann dieses Projekt als Erfolgreich eingestuft abgeschlossen werden.

Zuletzt sei noch erwähnt, dass eine genaue Recherche zu UI Frameworks hilfreich gewesen wäre und wir uns wahrscheinlich nicht für Swing, sondern für JavaFX entschieden hätten. Die Vorteile, welche man mit dem SceneBuilder und fxml hat, kann Swing nicht mitbringen.

1. Abbildungsverzeichnis

[Abbildung 1: Meilensteinplan 6](file:///C:\Users\schwa\Documents\GitHub\SE-Internship\documentations\G5-user%20-%20Copy.docx#_Toc110087714)

[Abbildung 2: erster Prototyp 6](file:///C:\Users\schwa\Documents\GitHub\SE-Internship\documentations\G5-user%20-%20Copy.docx#_Toc110087715)