TourPlaner_Andreas Documentation

App Architecture:

The Solution consists of 7 project which I am going to describe individually.

TourPlaner_andreas:

The main project and the project which gets started by visual studio. This Project holds the different Views, ViewModels and the Appconfig File.

The Appconfig File holds of the Database Connection string, Log4Net settings and other Settings like whether the Database connection should be used or a Local File System.

The views are responsible for the Graphical User Interface. The display the GUI and have minimalistic input validation functions.

The MainViiewModel interacts with all Data collected by the different views and is responsible to call the correct function in the Business Layer. In addition it updates the current which should be displayed to the GUI.

TourPlaner_andreas.BL

The business Layer is responsible for most Logical decisions. It decides when to use which function and has input validation functions. It also consists of a WebApi class to Filter content received by the MapQuest Api tool. The AppManager class within the BL is a singleton.

TourPlaner_andreas.DAL

The Data Access Layer defines all functions needed to retrieve wanted Data from the Sql Database or File service. In addition it has the interfaces of TourItemDataAccessObject and TourLogDataAccessObject.

TourPlaner.DataAccessLayer.PostgresSqlServer

Establishes the Database Connection and executes all the SQL-Statements. All these Statements are Prepared SQL-Statements to not allow SQL-Injection. After the execution of these functions the retrieved information's are passed to the BuisnessLayer.

TourPlaner.DataAccessLayer.FileAccess

Consists of all functions that interact with the local File System like, generate Report or, import/export. It also saves the TourpIc locally.

TourPlaner_andreas.Models

Defines the TourItem and LogItem so that the other systems know what these Items are.

TourPlanerAndreasTests

All Unit Tests are Located here.

Describe architectural, UX and library decisions:

The goal of the Layer Architecture is that each Layer has a different responsibility. In this case The TourPlaner_Andreas Layer interacts with the GUI, The Business Layer does the Logical decisions and the the Data Access Layer retrieves and saves data to the Database/Filesystem. TourPlaner_Andreas being the highest Level and DataAccessLAyer being the lowest Level meaning that the higher level has more Dependencies and accesses.

The design for the UI consists mainly of some Fields to display the Data and a Task Bar on Top to interact with the Tours and Logs. I decided to place all the buttons to interact with the Data into the TaskBar so that they appear more consistent.

The included Libraries help to interact with the Files and Database Access.

Describe implemented design Patterns:

IAppManager is a singleton and is retrieved by that AppmanagerFactory. This is done so it is possible to change between the DatabaseAccess with the BuisnessLAyer or the FileAccess. There can only ever exist one Instant of the AppManager so that Data does not get overwritten.

Describe Unit Testing decisions:

Since my biggest Problem was the right interaction with the FileSystem most Test interact with the FileSystem or BuisnessLayer

BuisnessLayerTests: Include Some Tests that check if a new Added Tour consists of empty strings.

FileAccessTests: Tests the functionality of the File Access Layer to check if the correct Files are retrieved and the functions work.

ReportsTests: Check the functions that are responsible to create the Reports. They mostly check if some strings are empty.

Unique Feature:

A functioning exe File was Created.

Tracked Time: 92H

Date	Time
18.03.2021	4H
25.03.2021	8H
30.03.2021	2H
04.04.2021	3H
06.04.2021	3H
16:04.2021	5H
17.04.2021	4H
20.04.2021	3H
15.05.2021	6H
16.05.2021	2H
17.05.2021	1H
26.05.2021	4H
02.06.2021	5H
03.06.2021	6H
04.06.2021	8H
05.06.2021	10H

06.06.2021	11H
07.06.2021	7H