Data Converter – Manual

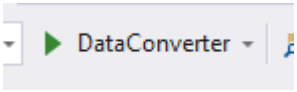
1. How to use DataConverter

1) Open the file DataConverter.sln
2) Update all filepaths in Program.cs at the top **if structure of folders changed**. I recommend using copies of original files. There are two ways of inserting filepaths:
    a. @"complete path as it comes from the Explorer"
    b. "complete filepath but with "\" replaced by "/"" (much slower)

    Filepaths as follows might work without setting if structure of folders not changed: e.g. `var filepathDB = Path.GetFullPath("../../../../../BuildingElementsKnowledge.db");`

    This means „../" means go one folder up from where you (= the program) are right now, which is in my case folderpath\DataConverter\DataConverter\bin\Debug\netcoreapp3.1

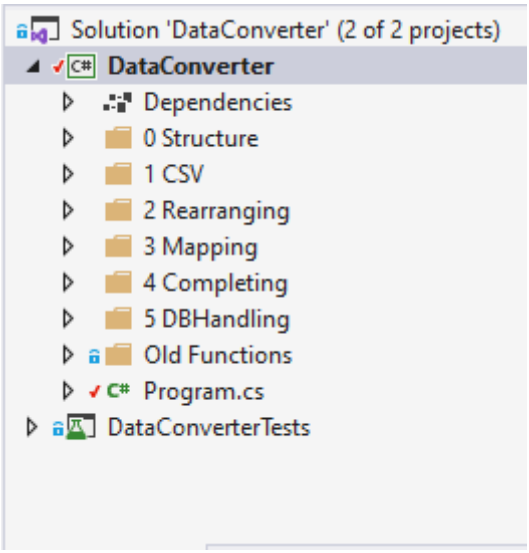
3) Always choose a backup copy from your db, if something messes up!
4) Press on run



2. Code Structure

The following explains the functions in the project DataConverter.

The project can more or less be subdivided in the following blocks.



0 Structure

Contains classes used throughout the program

- Constants and Desired Units
- StructureOekobaudat: Contains man structure of csv file (corresponds mainly to the first row of the csv file)
- SingleModEntry: Corresponds to one row in the Ökobaudat csv file (one row per UUID and module)
- OekobaudatEntry: Corresponds to all rows (= all modules) in the Ökobaudat csv file with the same UUID (=same material)

1 CSV

Contains functions that handle import from and export to csv files

- CSVHandler: functions for import
- OekobaudatCSVParser: functions to import specifically the StructureOekobaudat and List of SingleModEntries from the ökobaudat csv file
- CsvExportHandler: functions for csv export
- `OverviewKGLayerTypeMapping`: special function to export the mapping for all kgs and all layertypes (for checking purposes)

2 Rearranging

Contains different functions that in some way rearrange the existing data

- FilterHandler: functions that filter entries from (mostly) the List of all SingleModEntries
- SortingHandler: functions that sort List of SingleModEntries into different applications (most interesting is KG300)
- Order: functions that order given entries according to task into different groups (e.g. without duplicates etc; name not really suitable)
- MultipleHandler: functions that deal with multiple entries (several entries with same UUID but different module; needed for e.g. export and conversion to OekobaudatEntries)
- ConversionHandler: functions to convert from one type to another

3 Mapping

Contains functions concerned with the mapping of information (mostly application (functionality and position)) from external sources to entries

- CategoryMap: class with map from category to application
- UUIDMap: class with map from UUID to application
- ThermalConductivityMap: class with map from category to thermal conductivity
- MappingHandler: functions that use the classes above to map the information
- XmlHandler: adds/maps additional information from xml files of Ökobaudat

4 Completing

Functions to complete, correct and prepare all datasets for entering in the database

- DataCompletion: overview functions for the completion of different information of datasets (e.g. thickness, unit)
- CompletingHandlerSMEntry: functions specifically for completion of SMEntries (SingleModEntries)
- CompletingHandlerOekobaudatEntry: functions specifically for completion of OekobaudatEntries
- CorrectUnitHandler: Functions that define which units are “correct” for certain input
- CorrectionHandler: functions to correct datasets
- AirHandler: function that generates static and moving air layer (not in Ökobaudat, but needed)

5 PrepareDB

Functions to prepare and fill the given DB

- ClearDBHandler: function to clear given DbSet

- DBReadingHandler: Reads given information (like KG3xxNames) from the db
- GenerateUnits: generates the allowed units for the db
- DBWritingHandler: overview function to write classes/information into the db
- GenerateOekobaudatData: generates OekobaudatData to be filled in the db
- GenerateLayers: generates Layers to be filled and linked into the db

3. Explanations Input and Output Files

CSV files: Files that serve as input for the calculations in DataConverter

- Baustoffkonfigurationen_OBD_2020_II_modified.csv: edited file of the building material configurations, from which information on EndOfLife and densities etc. is obtained
- ManualThicknessCompletion: Completion of data sets that are in square meters and need the thickness for e.g. the U-value (not for data sets with other units!!!). Presumably separated to prevent "unnecessary" conversions)
- ManualDataCompletion: Completion of all data sets that are in the wrong unit
- ManualServiceLife0: Manual service life data for data records that contain 0 or no value from the building material configuration
- ManualServiceLife50: Manual service life data for data records that contain the maximum value 50 from the building material configuration
- ManualMappingUUID: all data records that must be mapped to a usage (KG3xx and LayerType) individually (based on the UUID) and not based on the category
- MappingKategorien-LayerType: Mapping of categories to usages (KG3xx and LayerType) whenever possible
- OBD_2020_II: Ecobaudat csv in the version of 2020_II - do not touch, strange things happen when changes are made
- ThermalConductivity: Assignments of the lambda values for all thermally active layers based on the categories

CSVoutput: Files that are output from the program for checking etc.

- AddedConversions: if a conversion unit is added within CompletingHandlerSMEntry.cs AddMissingConversions() (from reference flow name), then these appear in this file. However, this does not currently happen, so the file is usually empty
- EntriesNeedThicknessForUValue: OekobaudatEntries that require a thickness for the calculation of the U-value (i.e. thermally active layers that have no thickness)
- DataManualMappingUUIDToLayerType: Contains data records that require manual mapping to LayerType and cost groups (should generally be empty, except in the event of changes to the data, e.g. new eco-build version)
- DateNeedThicknessMult: Data records in sqm that could need a thickness for future developments (e.g. if you want to write the information to an IFC)
- DataEntriesAfterSortingB6Mult: Data records that are in B6 after sorting
- DataEntriesAfterSortingB7Mult: Data records that are in B7 after sorting
- DataEntriesAfterSortingKG300Mult: Data records that are in KG300 after sorting
- DataEntriesAfterSortingKG400Mult: Data records that are in KG400 after sorting
- DataNotConvertedEntriesMult: Data records that are not converted from SingleModEntries to OekobaudatEntries and therefore do not appear in the database

- UnitCorrection: File that documents all automated unit corrections → should be checked manually when changes are made to see whether errors have crept in
- DataEntriesWithLayerTypesUndKGs: File that has proven to be helpful for checking the data records (only for viewing, changes in it are not applied)
- KG-LayerTypeMaps: File to control the assigned LayerTypes in a KG
- LayerType-KGMaps: File to control the assigned KGs of a LayerType

Files that should be empty to guarantee a completely correct conversion (for the current application):

- EntriesNeedThicknessForUValue → adapt in ManualThicknessCompletion
- DataManualMappingUUIDToLayerType → adjust in ManualMappingUUID
- DataNotConvertedEntriesMult → needs more detailed analysis