13.01.2021

Ulrich Stark

OTH Amberg-Weiden

CSV File Importer

Dokumentation

Inhalt

[1 Grundlegendes 1](#_Toc61427061)

[2 Funktionen 2](#_Toc61427062)

[2.1 Importieren von CSV-Dateien 2](#_Toc61427063)

[2.2 Importieren von XML-Dateien 2](#_Toc61427064)

[2.3 Zusammenfügen 2](#_Toc61427065)

[2.4 Exportieren 3](#_Toc61427066)

[3 Grafische Anwendung 3](#_Toc61427067)

[3.1 Ansicht der importierten Dateien 5](#_Toc61427068)

[3.2 Vorschau-Tabelle 6](#_Toc61427069)

[3.3 Fenster zum Exportieren 7](#_Toc61427070)

[4 Programmierschnittstellen 8](#_Toc61427071)

# Grundlegendes

Das Projekt teilt sich in das „src“ und das „example“ Verzeichnis auf.

Im „src“-Verzeichnis befindet sich der Python-Quellcode für den CSV File Importer und einer grafischen Beispielanwendung, die mit dem Framework Tkinter erstellt wurde.

Das „example“-Verzeichnis erhält Beispieldateien zum Importieren. Diese teilen sich in CSV, XML und XSL Dateien auf, die teilweise von mir erstellt wurden. Den anderen Teil bilden die vorgegebenen Dateien der Aufgabe.

Entwickelt wurde mit dem Versionsverwaltungstool Git. Es steht als Git-Repository unter <https://github.com/ulrichstark/csv-importer> öffentlich verfügbar. Die Commit-Historie kann ebenfalls dort nachvollzogen werden.

Dem Projekt liegt außerdem noch diese Dokumentation als Microsoft-Word und PDF-Datei bei. Über das PowerShell Skript „generateDoc.ps1“ kann zu jedem Modul im „src“-Verzeichnis mithilfe von pydoc eine einfache Dokumentation generiert werden. Die Datei „uml-diagram.drawio“ stellt das Projekt für die Webseite draw.io dar, mit dem das UML-Diagramm für die Programmierschnittstellen in Kapitel 4 modelliert wurde.

Um die grafische Beispielanwendung oder das Konsolen-Testprogramm zu starten, müssen zuerst alle benötigten Python Module installiert sein. Dazu zählen die Module pandas, lxml, tkinter, chardet, csv, re und pandastable. Dann sollte mit dem Kommandozeilenbefehl „cd src“ das aktuelle Verzeichnis auf das Quellcode-Verzeichnis geändert werden. Um dann die grafische Beispielanwendung zu starten, reicht der Befehl „python gui.py“. Zum Starten des Konsolen-Testprogramms benutzen Sie bitte den Befehl „python main.py“.

Quelle für die benutzten regulären Ausdrücke ist die Webseite <https://regex101.com>. Dort wurden sie erstellt, getestet oder aus dort angebotenen Vorlagen abgeleitet.

# Funktionen

## Importieren von CSV-Dateien

Es können CSV-Dateien mit beliebigem Format importiert werden. Man spricht statt Format auch von dem Dialekt einer CSV-Datei. Dabei erkennt das Programm automatisch das Trenn- und Quotierungszeichen und die Kodierung einer Datei. Außerdem wird versucht, aus der ersten Datenzeile im Vergleich zu den anderen Zeilen zu schließen, ob es sich bei ihr um eine Überschriftenzeile handelt.

Dem Benutzer steht es dabei frei, diese Parameter im Nachhinein zu verändern. Zum Beispiel, wenn das Trennzeichen einer Datei falsch erkannt wurde und somit kein fehlerfreies Importieren gewährleistet werden kann.

## Importieren von XML-Dateien

Zusätzlich können auch XML-Dateien importiert werden. Dazu muss auch noch eine XSL-Datei (= Extensible Stylesheet Language) angeben werden, die diese zu importierende XML-Datei in das CSV-Format „transformieren“ kann.

Auch bei dieser Möglichkeit des Importierens wird die Dateikodierung und die speziellen Parameter des CSV-Formats automatisch erkannt. Die Ausgabe der XSL-Datei kann also ein beliebiges Trenn- und Quotierungszeichen wählen, sollte aber trotzdem ohne Fehler importiert werden können.

## Zusammenfügen

Jede importierte Datei (CSV oder XML) wird in eine interne Tabelle im Speicher des Computers zusammengefügt. Dabei spielt die Reihenfolge der importierten Dateien und die Spaltenanzahl eine wichtige Rolle. Der erste Import setzt mit seiner Anzahl der Spalten die benötigte Spaltenanzahl aller folgenden Imports fest. Sollte eine Datei importiert werden, die sich von dieser Anzahl unterscheidet, wird sie übersprungen und ein Fehler wird ausgegeben.

Für die Beschriftung der Spalten wird die erste gültige Überschriftenzeile eines Imports herangezogen. Wenn keine der importierten Dateien eine derartige Kopfzeile besitzt, kann das Programm aus den Zelleninhalten jeder Spalte auf den Datentyp schließen und setzt diesen mit einer fortlaufenden Nummer als Beschriftung.

Folgende Datentypen können mithilfe von regulären Ausdrücken erkannt werden: Geo-Koordinaten, E-Mail-Adressen, URLs, Datum kombiniert mit Uhrzeit, Datum, Uhrzeit, Dezimalzahlen, Ganzzahlen oder boolesche Ausdrücke. Sollte keiner dieser Datentypen zutreffen, wird der allgemeine Typ „Text“ angenommen.

## Exportieren

Die kombinierten importierten Dateien lassen sich daraufhin als CSV- und XML-Datei exportieren. Bei beiden Möglichkeiten müssen der Name, Pfad und Kodierung der zu exportierenden Datei ausgewählt werden.

Um eine CSV-Datei zu exportieren, muss zusätzlich noch ein Trenn- und Quotierungszeichen gewählt werden.

# Grafische Anwendung

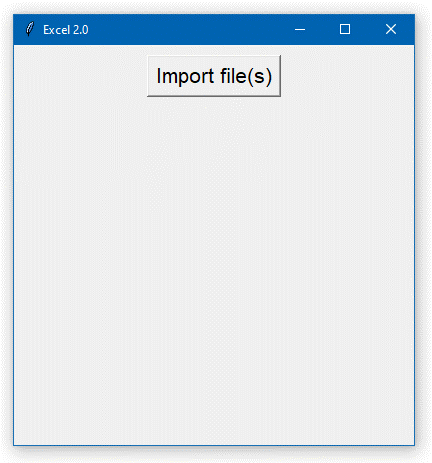
Die Grafische Anwendung dient als Beispiel für ein Programm, das die Programmierschnittstellen dieses Projekts benutzt. Entwickelt wurde sie mit dem GUI-Framework Tkinter, das in Python standardmäßig integriert ist.

Abbildung 1: Die Beispielanwendung direkt nach dem Starten

Nach dem Starten wird der Benutzer von einem einfachen Fenster auf seinem Desktop empfangen, das einen Knopf mit der Aufschrift „Import file(s)“ enthält. Andere Bedienelemente werden bewusst ausgeblendet bis der Nutzer Dateien zum Importieren ausgewählt hat. Dadurch wird er bei Programmstart nicht von unnötigen Eingabefeldern oder Knöpfen abgelenkt.

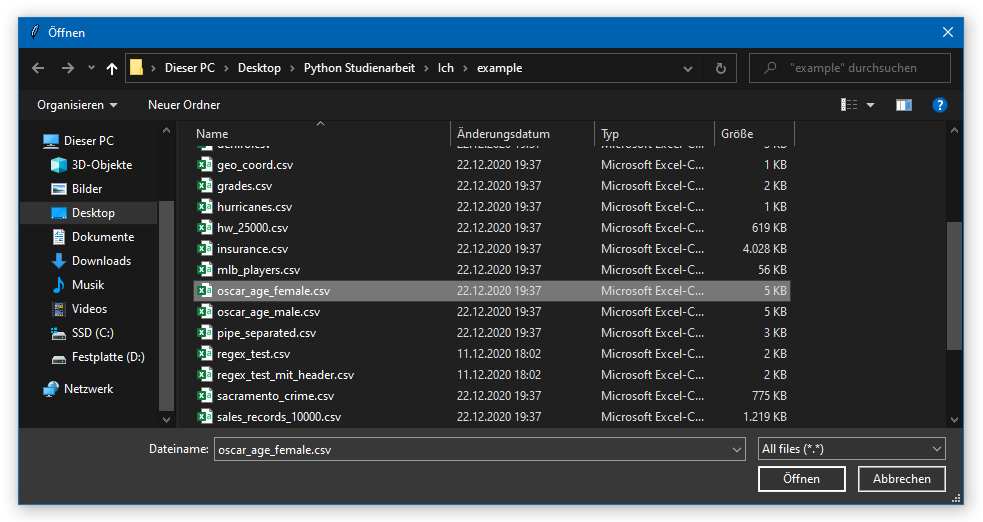
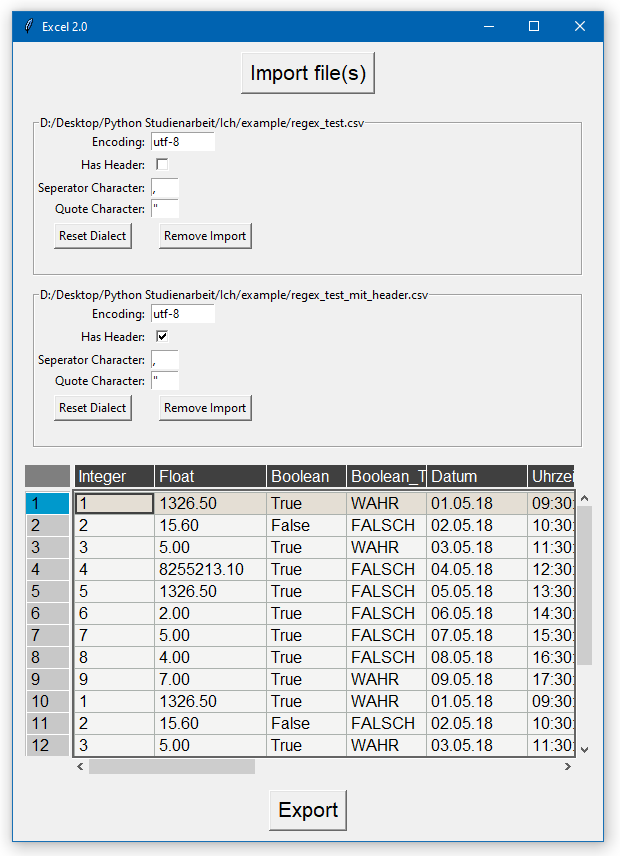
Wenn er den Knopf zum Importieren der Dateien klickt, erscheint der vom Betriebssystem bereitgestellte Dateiauswahl-Dialog.

Abbildung 2: Der Dateiauswahl-Dialog zum Importieren der CSV- und XML-Dateien

Dieser Dialog wurde gezielt so konfiguriert, dass er eine Mehrfachauswahl an Dateien zulässt. Außerdem wird standardmäßig das Verzeichnis mit den Beispieldateien geöffnet. Diese beiden Faktoren steigern die Effizienz beim Benutzen und Testen der Beispielanwendung.

Nachdem mindestens eine Datei importiert wurde, erscheinen die restlichen Bedienelemente. Das Fenster teilt sich nun in vier Bereiche.

Erstens in den Knopf zum Importieren von zusätzlichen Dateien. Darauffolgend ein Rahmen („Frame“) für jede importierte Datei. Drittens eine Tabelle, die als Vorschau für die zusammengefügten Dateien dienen soll. Und zuletzt ein Knopf mit der Aufschrift „Export“. Ein Klick auf diesen öffnet das Fenster zum Exportieren.

Abbildung 3: Ansicht der Beispielanwendung nach dem Importieren von zwei Dateien

## Ansicht der importierten Dateien

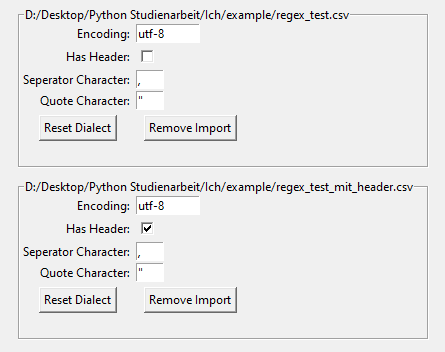
In diesem Teil der Anwendung erhält jede importierte Datei einen Bereich abgegrenzt durch einen Rahmen („Frame“), in dem ihre zugehörigen Parameter angepasst oder zurückgesetzt werden können. Außerdem steht ein Knopf zum Entfernen dieses Imports bereit.

Abbildung 4: Zwei importierte CSV-Dateien mit jeweils Parameter und Aktionen

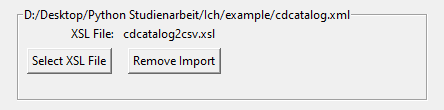
Der Frame für eine importierte CSV-Datei ermöglicht die Anpassung der Dateikodierung, des Trenn- und Quotierungszeichens und ob die Datei eine Kopfzeile besitzt. Diese Werte werden benutzt, um die Datei zu importieren und werden in den meisten Fällen automatisch richtig detektiert bevor der Nutzer eine Eingabe machen muss. Mit dem „Reset Dialect“-Knopf werden die Änderungen des Benutzers verworfen und die Werte abermals aus der Datei detektiert.

Abbildung 5: Importierte XML-Datei mit ausgewählter XSL-Datei

Im Frame einer importierten XML-Datei steht zusätzlich zu dem Entfernen-Knopf nur ein Knopf zum Auswählen einer XSL-Datei bereit. Dieses Dateiformat wird benutzt, um den Inhalt der XML-Datei in das CSV-Format zu überführen.

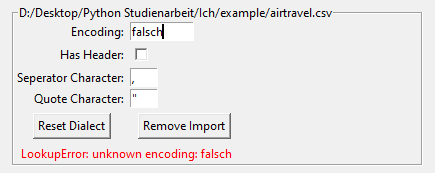
Wenn beim Importieren einer Datei ein Fehler auftritt oder die Parameter ungültig sind, wird ein Fehler in dem Frame ausgegeben. Dadurch wird der Benutzer nicht durch einen Fehlerdialog blockiert, sondern erhält eine Fehlerbeschreibung direkt in dem Kontext des Imports, in dem der Fehler aufgetreten ist.

Abbildung 6: Fehlermeldung bei falscher Angabe der Dateikodierung

## Vorschau-Tabelle

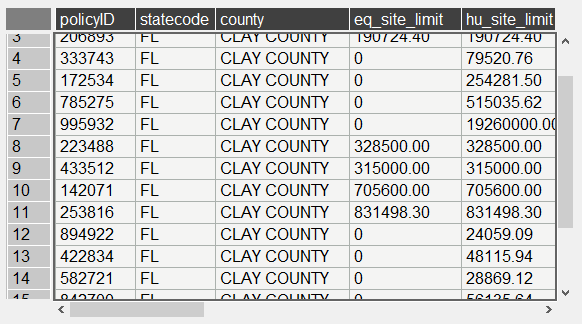
Die Tabelle unter den importierten Dateien dient dem Benutzer als Vorschau. Sie stellt dar, wie die zusammengefügten CSV- und XML-Dateien aussehen, wenn sie exportiert werden würden.

Abbildung 7: Vorschautabelle unter den importierten Dateien

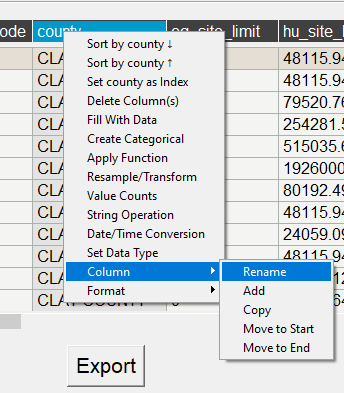
  
Außerdem kann der Benutzer damit zum Beispiel die Vorschau nach einer bestimmten Spalte auf- oder absteigend sortieren, die Reihenfolge der Spalten ändern oder über das „Column“ Untermenü einer Spalte eine neue Überschrift geben.

Abbildung 8: Kontextmenü einer Spalte für zusätzliche Aktionen

## Fenster zum Exportieren

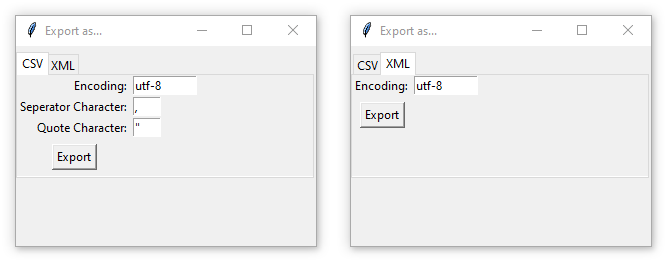
Nach einem Klick auf den „Export“-Knopf im Hauptfenster öffnet sich ein Unterfenster zum Exportieren der zusammengefügten Dateien. Dieses enthält eine Tableiste, mit der der Benutzer auswählen kann, ob er eine CSV- oder XML-Datei exportieren will.

Abbildung 9: Die zwei Zustände des Exportierfensters

Vor dem Export in eine CSV-Datei lässt sich die Kodierung der Datei und das Trenn- und Quotierungszeichen festlegen. Für den Export in eine XML-Datei steht nur das Ändern der Dateikodierung bereit.

Mit dem „Export“-Knopf im Unterfenster kann die Auswahl bestätigt werden und das Programm fragt den Benutzer mit dem Dateiauswahl-Dialog des Systems, wohin die Datei und mit welchem Namen sie gespeichert werden soll.

Wählt er einen gültigen Pfad und Namen aus, beginnt das Programm, die Datei zu exportieren und meldet dem Benutzer mit einem Dialog zurück, wenn der Export-Vorgang erfolgreich abgeschlossen wurde. Im Fehlerfall würde ein Fehlerdialog mit der relevanten Fehlermeldung erscheinen. Das könnte zum Beispiel eintreten, wenn der Benutzer sich für eine unbekannte Dateikodierung zum Exportieren entschieden hat.

# Programmierschnittstellen

Die Programmierschnittstellen für das Importieren, Zusammenfügen und Exportieren befinden sich in vier Python Module.

Um Dateien zu importieren, muss ein Objekt der Klasse Importer instanziiert werden. Für jede zu importierende Datei muss dann jeweils die Methode importCSVFile oder importXMLFile aufgerufen werden. Mit der reset-Methode kann man das Importer-Objekt zurücksetzen und mit dem Importieren neu beginnen.

Mit den „get“-Methoden können jederzeit während des Importierens die aktuell zusammengefügten Dateien entweder als Pandas DataFrame, NumPy Array, Liste an Listen oder Dictionary zurückgegeben werden.

Zum Importieren einer CSV-Datei wird ihr Dialekt benötigt. Dazu muss ein Objekt der Klasse Dialect an die Methode importCSVFile des Importer übergeben werden. Bei Instanziierung des Dialect Objekts werden seine Felder mit Standardwerten belegt. Wenn die Werte dieser Felder aus einer bestimmten Datei oder einem Beispielstring abgeleitet werden sollen, eignen sich die Methoden guessFromFile und guessFromSample der Dialect Klasse.

Um die zusammengefügten Daten zu exportieren, wird die Exporter Klasse benutzt. Ihr Konstruktor erwartet ein Objekt vom Typ Importer. Der Exporter stellt mit der Methode exportCSVFile eine Methode bereit, um eine CSV-Datei mit einem bestimmten Dialekt und Dateikodierung zu exportieren. Mit der Methode exportXMLFile kann eine XML-Datei mit der übergebenen Dateikodierung exportiert werden.

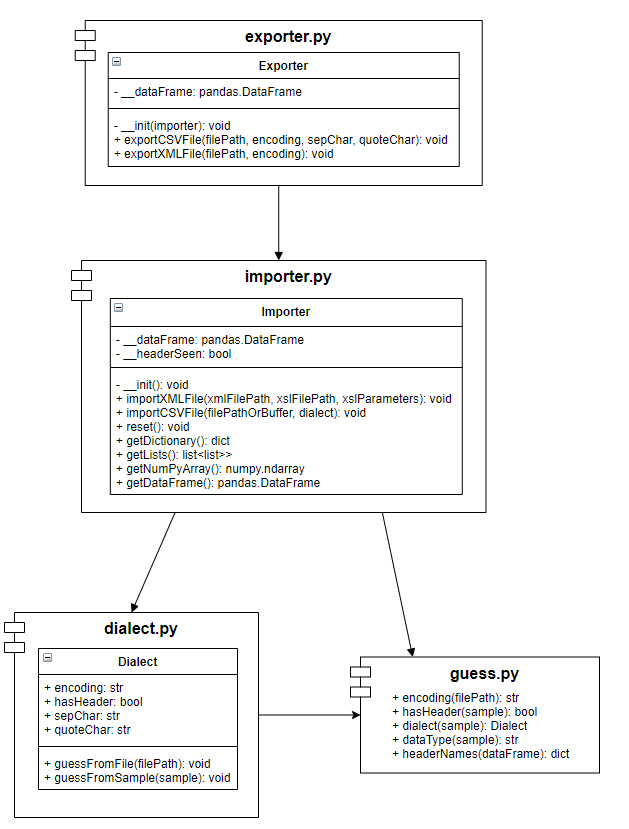
Die Methoden aus dem „guess“ Modul werden intern von der Importer und Dialect Klasse genutzt, können aber auch aus anderen Quellen aufgerufen werden. Mit diesem Modul kann erkannt werden, ob eine CSV-Datei eine Kopfzeile hat und mit welchem Dialekt oder Dateikodierung sie abgespeichert wurde. Außerdem stellt dieses Modul zwei Methoden bereit, um den Datentyp eines Zelleneintrags zu erraten und um darauf aufbauend Spaltenüberschriften für einen DataFrame zu generieren.

Abbildung 10: UML-Diagramm für Module der API