Kata "Encoding Converter"

Immer wieder steht man vor der Aufgabe ein oder mehrere Files von einer Kodierung in die andere Kodierung zu konvertieren.

# Aufgabe

Erstelle eine Command-Line Tool mit welchem ein einzelnes File wie auch mehrere Files auf einmal konvertiert werden kann.

# Anforderungen

Um sich eine bessere Vorstellung machen zu können, hier ein paar Beispiele wie das Tool in der Command-Line eingesetzt werden sollte:

JConverter ReadMe.txt –convertFrom ISO-8859-1 –convertTo UTF-8

Das ReadMe.txt File liegt im Zeichenformat ISO-8859-1 vor und soll in ein ReadMe.txt im Zeichenformat UTF-8 konvertiert werden. Das File liegt in dem Verzeichnis in dem der Konverter gestartet wurde.

JConverter \*.txt -convertFrom ISO-8859-1 -convertTo UTF-8

Konvertiert alle Files mit der Endung 'txt'.

Weitere (optionale) Möglichkeiten:

|  |  |
| --- | --- |
| \*.java | Matches a path that represents a file name ending in .java |
| \*.\* | Matches file names containing a dot |
| \*.{java,class} | Matches file names ending with .java or .class |
| foo.? | Matches file names starting with foo. and a single character extension |

JConverter \*.txt –d d:\data -r -convertFrom ISO-8859-1 -convertTo UTF-8

Konvertiert alles Files mit der Endung .txt die sich im Verzeichnis d:\data befinden. Mittels des Parameters -r (rekursiv) wird angegeben, dass auch alle Unterverzeichnisse durchsucht und allfällige vorhandene Files mit der Endung .txt auch konvertiert werden.

Mit der Angabe "UTF-8" ist "UTF-8 ohne BOM" gemeint.

Ein File bzw. File-Pattern muss immer angegeben werden. Die Argumente -convertFrom und ‑convertTo müssen immer angeben werden. Das Argument -r kann immer angegeben werden. Dem Argument ‑d kann eine Pfadangabe in "" folgen, z.B.: "c:\temp\test file".

Wird das Porgramm ohne Argumente oder mit fehlenden Argumenten gestartet, soll eine Hilfestellung auf die Console ausgegeben werden.

# Ziele

## Basisziele

* InputStream / OutputStream / Reader / Writer etc.
* java.nio (Java >= 1.7)
* Stream, Lamda (Java 8)
* Testing
  + Unit-Test ohne Files
  + TDD
* Single Responsibility Principle ([SRP](http://clean-code-developer.de/die-grade/orangener-grad/#Single_Responsibility_Principle_SRP))
  + Welche Aufgaben / Verantwortlichkeiten sind vorhanden
  + Klassendiagramm

## Variante

* Einsatz von third-party libraries
  + Apache Commons IO ([Doku](https://commons.apache.org/proper/commons-io/))
  + Apache Commons CLI ([Doku](https://commons.apache.org/proper/commons-cli/))