RCF-Datenformat

Technische Dokumentation

Stand 01.06.2012

1. Allgemeines

Diese Dokumentation beschreibt den Aufbau von RCF-Dateien. RCF-Dateien dienen der Bereitstellung von Daten für Programme, welche Daten in Tabellenform erwarten. Hierzu gehören z.B. Listengeneratoren. Insbesondere wird das RCF-Format von dem Listengenerator *Waimea Report* verarbeitet.

Im Gegensatz zu einer Textdatei, die die einzelnen Datensätze zeilenweise bereitstellt und die einzelnen Datenfelder durch einen Separator voneinander trennt, gestattet eine RCF-Datei die Gestaltung einer weitaus dynamischeren Datenstruktur. Prinzipiell kann in jedem Datensatz eine neue Feldstruktur definiert werden, bzw. die Menge der verfügbaren Datenfelder kann jederzeit während des Datenstromes dynamisch ergänzt werden.

Felder, die sich in mehreren aufeinanderfolgenden Zeilen inhaltlich gleichen, brauchen nur einmal definiert zu werden.

2. Dateiaufbau

RCF-Dateien sind zeilenweise organisierte Textdateien. Die Codierung erfolgt im ANSI-Code. Eine RCF-Datei muss mit der Zeile

WAIMEA-RCF-FILE

beginnen. Diese Zeile ist ein sogenannter *Magic*. Alle Programme, die eine RCF-Datei verarbeiten, prüfen nach dem Öffnen der Datei das Vorhandensein dieser Zeile und geben eine Fehlermeldung aus, das die Datei keine gültige RCF-Datei ist, wenn die Zeile nicht vorhanden ist.

Alle weiteren Zeilen der Datei enthalten Daten-, Befehls- oder Kommentarzeilen. Eine RCF-Datei darf beliebig viele Leerzeilen enthalten. Diese werden bei der Verarbeitung überlesen.

2.1. Kommentarzeilen

Kommentarzeilen beginnen mit einem doppelten Schrägstrich //. Alle Zeilen, die mit einem doppelten Schrägstrich beginnen, werden überlesen. Dabei ist es gleichgültig, welche und wie viele Zeichen sich hinter dem Schrägstrich in der Zeile befinden.

2.2. Befehlszeilen

Befehlszeilen beginnen mit einem Punkt. Hinter dem Punkt folgt ohne Zwischenraum der Name des Befehls. Die Syntax sieht Befehle mit oder ohne Parameter vor. In der augenblicklichen Definition sind folgende parameterlose Befehle definiert:

.DATA
.NEXT
.FORMFEED

Der Befehl **.DATA** leitet den eigentlichen Datenstrom ein. Alle Zeilen zwischen der Kopfzeile und dem Befehl **.DATA** enthalten globale Steueranweisungen. Folgende globale Steueranweisungen sind möglich:

.DECSEP

Die Syntax dieser Anweisung ist: .DECSEP=x

Sie definiert dasjenige Zeichen, das in Gleitkommazahlen als Dezimalkomma erkannt wird. Soll das Dezimalkomma ein Komma sein, so lautet der Befehl: .DECSEP=,

Neben dem Zeichen, das hier ausdrücklich als Dezimalkomma definiert wird, wird auch der Punkt als Dezimalkomma interpretiert.

.NOESCAPE

Die Syntax dieser Anweisung ist: .NOESCAPE=1

Wenn diese Anweisung auftritt, wird das Erkennen von Steuersequenzen, die mit einem Backslash \ beginnen, abgeschaltet. Es werden dann Sequenzen, wie \n, \t, usw. nicht mehr als Zeilenvorschub oder Tabulator interpretiert.

.ESCAPESYMBOL

Die Syntax dieser Anweisung ist: .ESCAPESYMBOL=60

Sie definiert dasjenige Zeichen, das als Fluchtsymbol in Steuersequenzen erkannt wird. Die Voreinstellung ist der Backslash \.

Die Angabe des neuen Zeichens erfolgt als ganze Zahl, die den Wert des Zeichens repräsentiert.

.BUFFSIZE

Die Syntax dieser Anweisung ist: .BUFFSIZE=30000

Wenn diese Anweisung auftritt, wird die Länge des internen Zeilenpuffers von der Voreinstellung 30000 auf die angegebene Anzahl Bytes hochgesetzt. Dies ist notwendig, wenn Zeilen in die Datei geschrieben werden, die länger als 30000 Bytes sind.

Der Befehl **.NEXT** beendet einen Datensatz. Er ist vergleichbar mit dem Zeilenende in einer zeilenweise organisierten Datendatei. Alle bis zum Befehl **.NEXT** definierten Datenfelder stehen dem verarbeitenden Programm als Felder der aktuellen Datenzeile zur Verfügung. Da alle Felder intern zwischengespeichert werden, stehen auch alle diejenigen Felder im Datensatz zur Verfügung, die in vorherigen Datensätzen definiert wurden.

Folgen beispielsweise zwei **.NEXT** Befehle direkt aufeinander, so entspricht das der Definition zweier identischer Datensätze.

Wenn der Befehl **.FORMFEED** vor der Definition eines Datensatzes oder zwischen den einzelnen Felddefinitionen eines Datensatzes eingefügt wird, wird für den betroffenen Datensatz eine neue Seite begonnen. Der Befehl muss auf jeden Fall vor dem Befehl **.NEXT** stehen, welcher den Datensatz beendet.

Jeder Datensatz **muss** durch einen Befehl **.NEXT** abgeschlossen sein. Dies gilt insbesondere für den letzten Datensatz in der Datei. Das Dateiende ersetzt den **.NEXT** Befehl nicht. Daher lautet der letzte Befehl einer RCF-Datei in der Regel **.NEXT**.

2.3. Datenzeilen

Alle anderen Zeilen sind Datenzeilen. Jede Datenzeile definiert ein Datenfeld und dessen Inhalt. Die Zeile beginnt mit dem Feldnamen. Dann folgt ein Gleichheitszeichen. Alle Zeichen hinter dem Gleichheitszeichen bis zum Zeilenende bilden den Feldinhalt. Datenzeilen, die kein Gleichheitszeichen enthalten, werden überlesen.

Aus obigen Syntaxregeln folgen insbesondere folgende Einschränkungen:

- Feldnamen können nicht mit einem Punkt beginnen
- Feldnamen können nicht mit einem doppelten Schrägstrich beginnen
- Feldnamen dürfen keine Gleichheitszeichen enthalten

Der Inhalt eines Datenfeldes kann einer der folgenden Datentypen sein:

- Text
- Ganze Zahl
- Gleitkommazahl
- Datum
- Uhrzeit

Dabei sind folgende Syntaxregeln zu beachten:

- Ganze Zahlen dürfen nur Ziffern oder die Vorzeichen + oder – enthalten.
- Gleitkommazahlen dürfen nur Ziffern, Vorzeichen oder einen Punkt als Dezimalkomma enthalten.
- Datumsangaben werden in der Reihenfolge *Tag*, *Monat*, *Jahr* angegeben. Die Komponenten sind durch einen Punkt voneinander zu trennen. Das Jahr sollte vierstellig angegeben werden. Zweistellige Jahresangaben werden mit 1900 ergänzt.
- Uhrzeitangaben werden in der Reihenfolge *Stunde*, *Minute*, *Sekunde* angegeben. Die Komponenten sind durch einen Doppelpunkt voneinander zu trennen. Die Stunden können im 24-Stunden-System angegeben werden.

Die Länge der Datenzeilen ist prinzipiell nicht begrenzt.

3. Beispiel

Die folgende Datei definiert einen Datenstrom von drei Datensätzen. Alle Datensätze stellen die Felder *Name*, *Vorname*, *Straße*, *Plz* und *Ort* sowie die Felder *Artikel*, *Menge*, und *Preis* zur Verfügung. Die Felder *Name*, *Vorname*, *Straße*, *Plz* und *Ort* werden jedoch nur ein einziges Mal definiert, so dass ihr Inhalt in allen Datensätzen der gleiche sein wird.

```
WAIMEA-RCF-FILE

.DECSEP=,

.DATA
Name=Schmidt
Vorname=Heinrich
Strasse=Datenweg 12
Plz=44799
Ort=Bochum
Menge=10
Artikel=Fenster
Preis=123,45
.NEXT

Menge=1
Artikel=Tür
Preis=333,33
.NEXT

Menge=3
Artikel=Dachfenster
Preis=434,00
.NEXT
```

Die oben beschriebene Beispiel-RCF-Datei stellt die selben Datensätze bereit, wie folgende zeilenweise organisierte Textdatei, die die Feldnamen in der ersten Zeile benennt und in der die Felder durch Semikolons getrennt sind:

```
Name;Vorname;Straße;Plz;Ort;Menge;Artikel;Preis
Schmidt;Heinrich;Datenweg 12;44799;Bochum;10;Fenster;123.45
Schmidt;Heinrich;Datenweg 12;44799;Bochum;1;Tür;333.33
Schmidt;Heinrich;Datenweg 12;44799;Bochum;3;Dachfenster;434
```

