RPG and Unicode – Teil 3
Hardcodierte Werte und Prüfung Datenverlust bei CCSID-Konvertierung

Bereits seit Release 7.2 ist es möglich RPG-Variablen mit einer CCSID zu definieren und somit auch eine Konvertierung zwischen den verschiedenen CCSIDs zu erreichen. Was passiert jedoch mit hardcodierten Werten, für die keine CCSID angegeben werden kann? Gibt es außerdem eine Möglichkeit CCSID-Konvertierungen, die zu Datenverlust führen könnten, festzustellen? Beide Fragen werden in diesem Artikel beantwortet.

Hardcodierte Werte, Literale und Compile-Time-data

Hardcodierungen, insbesondere die Hardcodierung von Sonderzeichen sollte in Quellen soweit möglich vermieden werden. Nur leider ist das ist das nicht immer möglich. Werden z.B. Regular Expressions verwendet, sind für die Patterns Sonderzeichen erforderlich. Ähnliches gilt, wenn JSON-Daten direkt im RPG-Programm aufbereitet werden.

Die Frage ist nun, mit welcher CCSID werden diese Literale verarbeitet?

Wird ein Modul erstellt, wird vom Compiler per Default die CCSID der Quelle verwendet, um die Quelle zu lesen und das Objekt zu erstellen. Die CCSID kann jedoch im Compile-Command CRTRPGMOD oder CRTBNDRPG über die Option TGTCCSID geändert werden.

Gehen wir davon aus, dass die CCSID der Quelle verwendet wird, z.B. CCSID 273 (= Deutsch). Die Vermutung liegt nahe, dass die hardcodierten Werte in dieser CCSID im Modul und damit im (Service-)Programm hinterlegt und verwendet sind. Leider ist dem nicht so!

Wird das (Service-)Programm ausgeführt werden die hardcodierten Werte per Default in die CCSID des Jobs konvertiert. Solange es sich bei den hardcodierten Werten lediglich um (lateinische) Buchstaben und Ziffern handelt ist das auch kein Problem. Sollten jedoch Sonderzeichen hardcodiert sein, z.B. weil Regular-Expressions verwendet werden, kann dies zu Problemen führen.

Gehen wir davon aus, dass in einem Programm Regular-Expressions verwendet wurden, z.B. zum Prüfen von e-Mail-Adressen. Zur Definition des Patterns für Regular Expression müssen Sonderzeichen verwendet werden. Mit dem Pattern ^\w+([.-]?\w+)*@\w+([.-]?\w+)*(\.\w{2,3})+$ kann z.B. eine e-Mail-Adresse geprüft werden.

Das Programm konnte auf dem deutschen System mit der Quellen-CCSID 273 korrekt umgewandelt werden. Das Programm läuft auch korrekt in einem Job mit CCSID 273 oder 1141. Jetzt soll dieses Programm jedoch z.B. in der kyrillischen Umgebung ausgeführt werden. Da per Default auch die hardcodierten Werte zur Laufzeit in die CCSID des Jobs konvertiert werden und die Sonderzeichen im deutschen Zeichensatz einen anderen Hex-Wert als im Kyrillischen haben, funktionieren unsere Regular Expressions nicht mehr.

*Schlüssel-Wort CCSID(*EXACT) in den Control-Options (H-Bestimmungen)*

Abhilfe kann an dieser Stelle das Schlüssel-Wort CCSID schaffen, das in den Control-Options (H-Bestimmungen) angegeben wird.

Über das Schlüssel-Wort CCSID können die Default-CCSIDs für alphanumerische Single- und Double Byte Variablen, die ohne CCSID definiert werden, festgelegt werden.

Diese Defaults gelten jedoch nicht für hardcodierte Werte. Für die hardcodierten Werte kann jedoch zusätzlich in den Control-Options (H-Bestimmungen) das Schlüssel-Wort CCSID(*EXACT) angegeben werden. Das bedeutet, dass zur Laufzeit keine CCSID-Konvertierung erfolgt und die hardcodierten Werte in der CCSID der Quelle bzw. in der CCSID, die in Option TRGCCID im Compile-Befehl angegeben wurde, verwendet werden. Damit können auch die Regular-Expressions, in der kyrillischen Umgebung korrekt ausgeführt werden.

CCSID-Konvertierung kann zu Datenverlust führen

Bei einer CCSID-Konvertierung (z.B. der Übernahme von UTF-8-Daten oder Double Byte Daten in EBCDIC-Variablen) kann es zu Datenverlust kommen, da nicht unterstütze Zeichen durch Substitutions-Zeichen (und zwar immer das gleiche) ersetzt werden. Normalerweise tritt an dieser Stelle in RPG kein Fehler auf.

RPG and Unicode – Teil 3
Hardcodierte Werte und Prüfung Datenverlust bei CCSID-Konvertierung

Es stellt sich nun die Frage, ob es in RPG eine Möglichkeit gibt, festzustellen ob und welche Daten nicht vollständig konvertiert werden konnten.

RPG-Status +50

Sofern bei einer alphanumerischen Konvertierung Substitutions-Zeichen ausgegeben werden, wird der RPG-Status auf +50 gesetzt. In diesem Fall tritt zwar kein Fehler auf, aber der Status kann mit Hilfe der Built-In-Funktion %STATUS geprüft werden.

In dem folgenden Beispiel werden UTF-8-Daten in Variable DSRow.Text in eine EBCDIC-Variable (mit CCSID 273 = Deutsch ohne Euro) übertragen. Anschließend wird geprüft (%STATUS() = 50), ob bei dieser Konvertierung Substitutions-Zeichen ausgegeben wurden.

```
LocText273 = DSRow.Text;
If %Status() = 50;
   DSPLY 'Data get lost - Substitution Characters';
EndIf;
```

Abbildung 1: Built-In-Function %STATUS - Prüfen Datenverlust bei der Konvertierung CCSID

Nun ist es müßig nach jedem Statement bei dem eine CCSID-Konvertierung stattfinden könnte, den Status 50 abzufragen. Des Weiteren ist es auch nicht immer ersichtlich, ob und wo genau eine CCSID-Konvertierung erfolgt. Dieser Fall tritt z.B. auf, wenn die Texte mit unterschiedlichen CCSIDs verknüpft und konvertiert werden.

Schlüssel-Wort CCSIDCVT in den Control Options (H-Bestimmungen)

Diesem Problem kann man mit Hilfe des Schlüssel-Wortes CCSIDCVT, das bei den Control-Options (H-Bestimmungen) angegeben wird, abhelfen.

Das Schlüssel-Wort CCSIDCVT hat zwei Ausprägungen

- *EXCP — Wird *EXCP in Schlüssel-Wort CCSIDCVT angegeben, erfolgt zur Laufzeit ein Programm-Abbruch, sofern bei der CCSID-Konvertierung Daten verloren gehen also Substitutions-Zeichen ausgegeben werden.
- *LIST — Wird *LIST in Schlüssel-Wort CCSIDCVT angegeben, wird zur bei der Kompilierung ein zusätzlicher Abschnitt in die Kompilierungsliste eingefügt.

  In diesem Abschnitt werden alle CCSID-Konvertierungen aufgelistet, incl. Von- und Nach-CCSID, sowie die Quellen-Zeilen bei denen die Konvertierung erfolgt.

  Der zusätzliche Abschnitt wird jedoch nur eingefügt, wenn keine Kompilierungs-Fehler aufgetreten sind und wenn das Schlüssel-Wort OPTIONS(*NOGEN) nicht angegeben wurde.

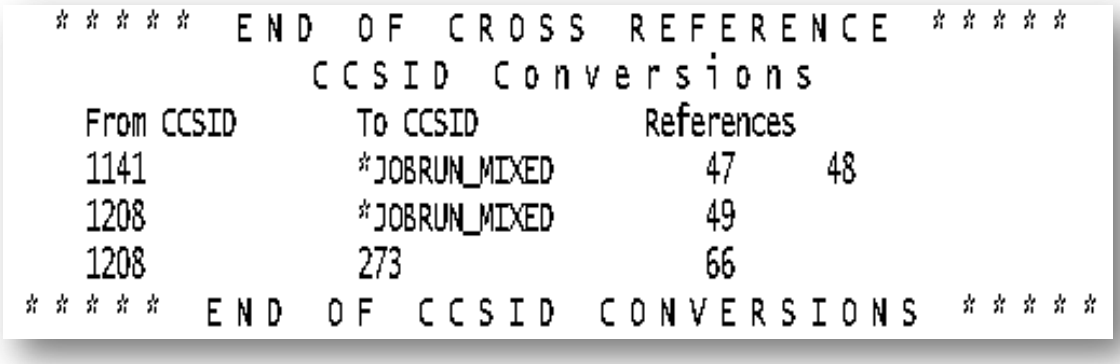
Beide Ausprägungen können gleichzeitig angegeben werden, also CCSIDCVT(*LIST: *EXCP) oder CCSIDCVT(*EXCP: *LIST).

Das folgende Beispiel zeigt den Abschnitt der CCSID-Konvertierungen aus der Kompilierungsliste eines Programms, bei dem CCSIDCVT(*LIST) angegeben wurde.

In Zeile 47/48 erfolgt eine CCSID-Konvertierung von CCSID 1141 (Deutsch mit Euro) in die Job CCSID. In Zeile 48 erfolgt eine CCSID-Konvertierung von UTF-8 (CCSID 1208) in die Job CCSID. In den Zeilen 47-49 wird die Ein-/Ausgabe-Struktur der in den F-Bestimmungen angegebenen Datei definiert. Per Default werden die alphanumerischen Single-Byte-Spalten (unabhängig davon mit

welcher CCSID die Spalten in der Tabelle definiert wurden) in die CCSID des Jobs konvertiert. Zumindest in Zeile 49 kann es somit zur Laufzeit zu Konvertierungs-Problemen kommen, da die Spalte in der Tabelle mit UTF-8 definiert wurde.

In Zeile 66 erfolgt eine weitere CCSID-Konvertierung von UTF-8 in CCSID 273 (Deutsch ohne Euro). Auch an dieser Stelle können zur Laufzeit Konvertierungs-Probleme auftreten.



```
*****  E N D  O F  C R O S S  R E F E R E N C E  *****
                C C S I D  C o n v e r s i o n s
   From CCSID       To CCSID          References
   1141             *JOBRUN_MIXED       47     48
   1208             *JOBRUN_MIXED       49
   1208             273                 66
*****  E N D  O F  C C S I D  C O N V E R S I O N S  *****
```

Abbildung 2: CCSID Konvertierungs-Abschnitt in der Kompilierungs-Liste - Schlüssel-Wort CCSIDCVT(*LIST)

Mit Hilfe der CCSID-Konvertierungs-Informationen, können unnötige Konvertierungen entdeckt und die CCSIDs der Variablen entsprechend angepasst werden. Weniger Konvertierungen bedeuten auch eine bessere Performance.

Zum zweiten können die Konvertierungen, bei denen es zu Datenverlust durch Substitutionszeichen kommen kann (z.B. von 1208 nach 273) festgestellt werden. Mit einer entsprechenden Korrektur des Source Codes (z.B. Variablen in 1208 statt 273) können potenzielle Konvertierungs-Probleme weitgehend ausgeschlossen werden.

Fazit

Um Problemen mit bei der Konvertierung zwischen unterschiedlichen CCSIDs in Variablen oder Problemen bei unterschiedlichen CCSIDs in der Entwicklungs- und Laufzeit-Umgebung weitgehend zu entgehen, sollten die folgenden Einträge in den Control-Options (H-Bestimmungen) hinzugefügt werden:

- CCSIDCVT(*EXCPT: *LIST): Durch die Angabe von *LIST wird in die Kompilierungsliste ein zusätzlicher Abschnitt eingefügt, über den alle CCSID-Konvertierungen aufgelistet sind.

  Die Angabe *EXCPT bewirkt, zur Laufzeit einen Abbruch erfolgt, wenn bei der Konvertierung zwischen CCSIDs nicht alle Zeichen umgesetzt werden können.

- CCSID(*EXACT) Bewirkt, dass hardcodierte Werte zur Laufzeit nicht in die aktuelle Job CCSID konvertiert werden.

Soweit zur Konvertierung von hardcodierten Werten und wie man Datenverlust durch CCSID-Konvertierung feststellen kann.

Im nächsten Artikel werden wir uns mit der Verarbeitung von Tabellen mit Spalten in unterschiedlichen CCSIDs mit native I/O beschäftigen.

Bis dahin schon einmal viel Spaß beim Konvertieren und Verarbeiten von alphanumerischen Daten in unterschiedlichen CCSIDs.