Kapitel 6 – Syntax REXX

Instruktionen

Bei den REXX-Instruktionen unterscheidet man zwischen

- Instruktionen, die der REXX-Sprachinterpreter ausführt
- Instruktionen, die zur Ausführung an das TSO weitergeleitet werden (extended TSO-Functions)
- TSO-Kommandos, beziehungsweise Anweisungen an eine andere eingestellte oder adressierte Umgebung

Die beiden letzten werden als Literale kodiert. Enthalten diese Instruktionen Variablen, sind die Anführungszeichen vor der Variablen zu beenden und nach der Variablen fortzuführen, da anderenfalls die Variable nicht als solche erkannt werden kann.

Funktionen

- Funktionen sind Bestandteile von Instruktionen. Zur Laufzeit tritt an die Stelle der Funktion der Wert, den sie retourniert.
- Eine Funktion wird vom Interpreter daran erkannt, dass dem Namen der Funktion immer unmittelbar ein Klammernpaar folgt. Sollen Informationen an die Funktion übergeben werden, sind diese innerhalb der Klammern anzugeben.
  SAY TIME() /* Gibt die aktuelle Uhrzeit aus */
- Eine Funktion, die alleine in einer Zeile steht (außerhalb einer Instruktion) wird vom REXX-Interpreter mit einem RC-3 zurückgewiesen. Zuweilen tut dies der Wirksamkeit keinen Abbruch (extended TSO-Functions). Der besseren Lesbarkeit willen sollte auf diese Schreibweise aber generell verzichtet werden.

REXX kennt über 50 sogenannter Built-in-Funktionen, die sich in unterschiedliche Kategorien einteilen lassen:

- Arithmetische Funktionen
- Vergleichsfunktionen
- Funktionen zum Konvertieren von Daten
- Formatierungsfunktionen
- Funktionen zur String-Manipulation
- Nicht zuordenbare restliche Funktionen

Darüber hinaus sind die erweiterten TSO-Funktionen verfügbar.

Hinweis Syntaxbeschreibung

Das Buch erklärt nicht alle TSO-Kommandos und auch nicht jeden Operanden, sondern nur diejenigen, welche in Verbindung mit REXX häufig benutzt werden. Ich wollte die 1000-Seiten-Grenze nicht überschreiten. Für eine Detailinformation lesen Sie bitte im Manual "TSO Command Reference" nach.

Die REXX-Instruktionen, Built-in-Functions, TSO-Kommandos und external TSO Functions werden in alphabetischer Reihenfolge beschrieben.

Ich bitte um Verständnis, dass nicht jeder einzelne Operand bis ins Detail angeführt und erklärt wird. Das vorliegende Buch hätte sonst eine vierstellige Seitenzahl. Details sind zu finden in TSO/E REXX Reference.

ABBREV() – Built-in-Function

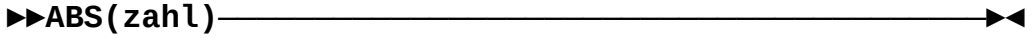
Übergibt eine 1, wenn string2 identisch mit dem Beginn von string1 ist und die Länge von string2 nicht größer der angegebenen Länge ist. Wird diese Bedingung nicht erfüllt, erfolgt die Rückmeldung von 0.

Beispiele

```
SAY ABBREV('Print','Pri')       1
SAY ABBREV('PRINT','Pri')       0
SAY ABBREV('PRINT','P',1)       1


/**REXX*****************************/
PARSE UPPER ARG TEST .
IF ABBREV(TEST,'T',1) THEN TRACE '?R'
```

ABS() – Built-in-Function

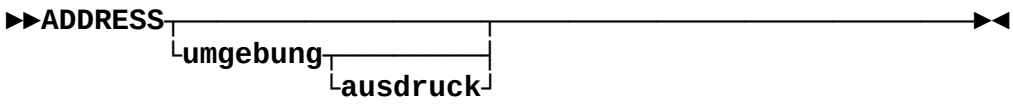


Übergibt den absoluten Wert einer Zahl ohne Rücksicht auf das Vorzeichen und führende Nullen.

Beispiele

```
SAY ABS('005.95')     5.95
SAY ABS('-6.04')      6.04
```

ADDRESS – REXX instruktion



Mit ADDRESS wird die Systemumgebung benannt, an die ein Befehl zu übergeben ist. Erfolgt keine Befehlsangabe hinter der Umgebung, wird das angegebene Environment zum Standard, bis eine erneute Änderung erfolgt.

Umgebung

Unter TSO/E REXX sind folgende Umgebungen durch den ADDRESS-Befehl erreichbar:

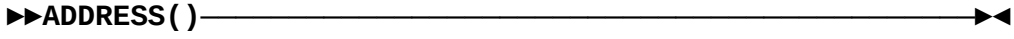
TSO	TSO-Befehlsinterpreter
LINK	das LINK-Environment
ISPEXEC	ISPF-Dialog Manager
ISREDIT	ISPF-Editor (für Makroprogrammierung)

Beispiele

```
ADDRESS ISPEXEC "DISPLAY PANEL(PANEL1)"
```

```
ADDRESS ISPEXEC
"EDIT DATASET("varname")"
if RC = 0 then do
     "DISPLAY PANEL(PANEL1)"
     :
```

ADDRESS() – Built-in-Function



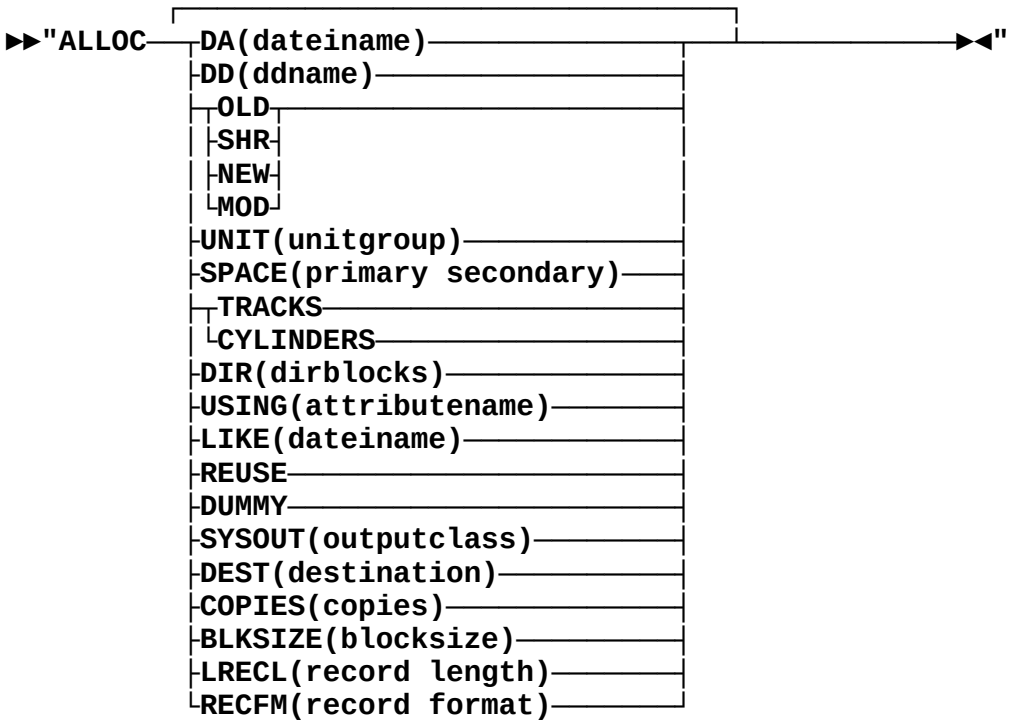
Meldet den Namen der Systemumgebung, innerhalb derer die Abarbeitung der externen Befehle durchgeführt wird.

Beispiele

```
SAY ADDRESS()          'TSO'      /* möglich */
SAY ADDRESS()          'MVS'      /* möglich */
SAY ADDRESS()          'ISPEXEC'  /* möglich */
```

```
:
OldEnv = ADDRESS()
Address TSO
Befehl
Befehl
Address OldEnv
:
```

ALLOC – TSO Kommando



Dient der Zuordnung von Dateien, die für ein auszuführendes Programm benötigt werden.

DA(dateiname)

Benennt die Datei, die unter einem bestimmten symbolischen Namen zugeordnet werden soll. Sollen mehrere Dateien unter einem symbolischen Namen verkettet werden (DATASET-CONCATINATION), können alle der Reihe nach innerhalb der runden Klammern angegeben werden. Dabei werden die Zugriffe der Reihe nach entsprechend der Schreibweise von links nach rechts durchgeführt. Bei Dateiverkettung ist darauf zu achten, dass die Datei mit der größten Blockgröße als Erste genannt wird.

Wird an Stelle eines Dateinamens Stern(*) angegeben, wird für den symbolischen Namen das Terminal zugeordnet.

DD(ddname)

Symbolischer Name, der in einem Programm oder einer Prozedur benutzt wird. An Stelle von FI ist auch DD möglich.

OLD

Die Datei ist bereits vorhanden und wird für die Dauer der Verarbeitung exklusiv belegt.

SHR

Die Datei ist bereits vorhanden. Während der Verarbeitung können auch andere Benutzer auf diese Datei zugreifen.

NEW

Die Datei wird eingerichtet. NEW beinhaltet das Katalogisieren.

MOD

MOD prüft, ob die Datei vorhanden ist. Wenn ja, wird die existente benutzt. Ist sie nicht vorhanden, wird sie angelegt.

Wird mit Disposition MOD in eine Datei geschrieben, werden die Daten an das Ende angefügt (Dateifortschreibung). MOD darf nicht für Member von PO-Dateien verwendet werden.

UNIT(...)

Beschreibt die Gerätegruppe der Datei. Zu beachten sind die unternehmensspezifischen Regeln unter SMS. (Nur bei Neuanlage)

SPACE(primary secondary)

Legt die Menge der Primärzuweisung(x) und gegebenenfalls der Sekundärzuweisung(y) fest.

TRACKS oder CYLINDERS

Nimmt Bezug auf die Größenangaben im SPACE-Operanden.

DIR(...)

Anzahl der DIRECTORY-Blöcke bei Erstellung einer PO-Datei. Jeder DIR-Block kann etwa 5 bis 6 Membereinträge aufnehmen.

USING(attr)

Ordnet die Dateiattribute zu, die in einem ATTR-Befehl unter dem angegebenen Namen definiert wurden. Dies stammt aus der frühen Vergangenheit und wird heute

kaum noch benutzt. Die Dateiattribute können im ALLOC auch direkt angegeben werden.

LIKE(dateiname)

Übernimmt alle Dateiattribute der angegebenen Datei.

REUSE

Überschreibt die Zuweisung eines symbolischen Namens in der Dateizuweisungstabelle. REUSE darf nur mit Disposition SHR erfolgen.

DUMMY

NOP-Anweisung (Null-Operation, Blindzuweisung).

SYSOUT(...)

Beschreibung einer Druckausgabedatei (SPOOLOUT). Die Druckausgabeklasse muss angegeben werden (A-Z und 0-9).

DEST(...)

Benennt den Remotedrucker, auf den die Druckausgabe geleitet werden soll.

COPIES(...)

Anzahl der Gesamt-Druckausgaben.

BLKSIZE(...)

Blockgröße der zu erstellenden Datei.

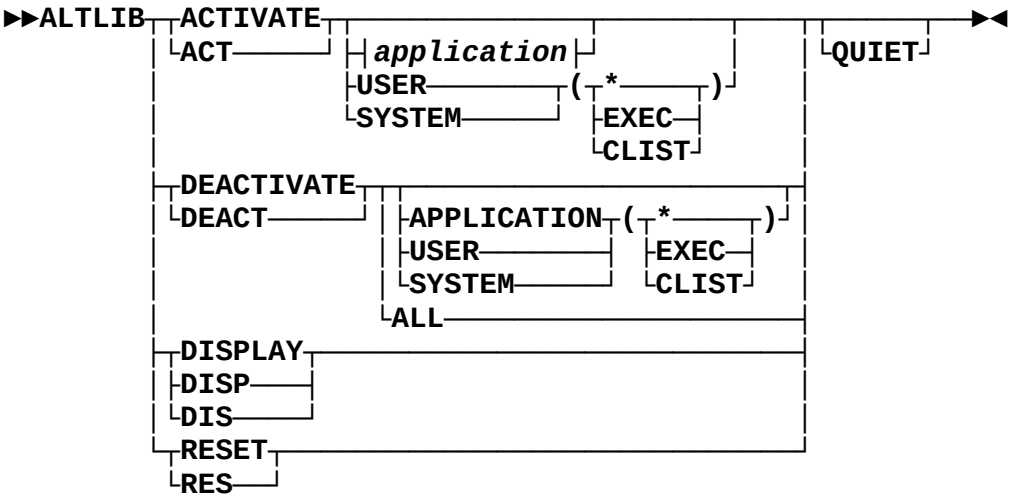
LRECL(...)

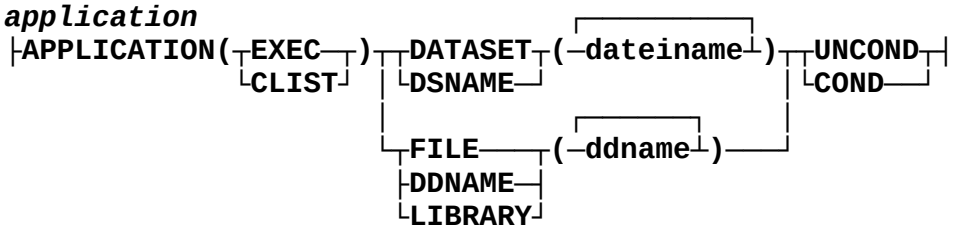
Satzlänge der Datei

RECFM(...)

Satzformat der Datei.

ALTLIB – TSO-Kommando





Das ALTLIB-Kommando kann für folgendes benutzt werden:

- Definition alternativer Applikations-Bibliotheken für REXX oder CLIST
- Definition Benutzer-spezifischer Bibliotheken für REXX oder CLIST
- Ausschluss einiger Bibliotheken bei der Suche nach REXX oder CLIST
- Rücksetzen der Definitionen.
- Anzeige der aktuellen Suchreihenfolge

ACTIVATE

Bestimmt, dass die Bibliothek in die Suchreihenfolge für REXX/CLIST einbezogen wird.

DEACTIVATE

Die Bibliothek soll bei der Suche nach REXX/CLIST nicht mehr berücksichtigt werden.

DISPLAY

Zeigt Informationen über die Suchreihenfolge an.

RESET

Setzt die Definitionen für Suchreihenfolge auf den SYSTEM-Level zurück.

USER

USER-Level für die Suche. Dieser wird über die DD-Namen SYSUEXEC/SYSUPROC zugewiesen.

APPLICATION

Der Application-Level wird durch die Zuweisung von Datei- oder DD-Namen zugewiesen.

SYSTEM

Die Definition des SYSTEM-Levels erfolgt über SYSPROC (für REXX und CLIST) oder SYSEXEC (nur für REXX).

ALL

Legt fest, dass alle Levels (SYSTEM, APPLICATION und USER) für REXX und CLIST deaktiviert werden. Möglicherweise hilft dann nur ein erneuter LOGON.

(EXEC)

Bestimmt, dass der entsprechende Level für REXX aktiviert oder deaktiviert wird.

(CLIST)

Bestimmt, dass der entsprechende Level für CLIST aktiviert oder deaktiviert wird.

(*)

Bestimmt, dass der entsprechende Level für REXX und CLIST aktiviert oder deaktiviert wird.

DATASET(datei) | DSNAME(datei)

Bestimmt, dass ein APPLICATION-Level definiert wird. Hier gibt es folgendes zu beachten:

- Die Dateien müssen PO- oder PO/E-Format haben

- Die Anzahl der Dateien ist auf 15 limitiert (DDNAME kann mehr)
- Die Dateien können unterschiedliche Blockgrößen aufweisen und in beliebiger Reihenfolge zugewiesen werden.
- Alle Dateien müssen das gleiche Satzformat haben.
- Membernamen dürfen nicht angegeben werden.

FILE | DDNAME | LIBRARY

Bestimmt, dass ein APPLICATION-Level definiert wird. Hier gibt es folgendes zu beachten:

- Der DD-Name muss vor Nutzung in ALTLIB zugewiesen sein.
- Der DD-Name muss mit einer PERMANENT-Attribut zugewiesen sein. Dateien, die während der LOGON-Phase zugewiesen werden, haben dieses Attribut automatisch.

UNCOND

Aktiviert den Application-Level auch dann, wenn bereits ein Application-Level vom gleichen Typ aktiv ist. Bis zu 8 Definitionen können gestacked werden.

COND

Aktiviert den Application-Level nur, wenn keiner vom gleichen Typ aktiv ist. Anderenfalls wird ein RC ^= 0 ausgegeben.

QUIET

Unterdrückt bis zu 99 Meldungen. Auf diese Weise kann ein ALTLIB-Display erfolgen und die Information im Programm verarbeitet werden. Siehe hierzu das folgende Beispiel

```
/* REXX
*/
ADDRESS TSO "ALTLIB DISPLAY QUIET"
ADDRESS ISPEXEC "VGET (IKJADM IKJADM1 IKJADM2 IKJADM3…)
SHARED"
SAY "IKJADM " IKJADM
SAY "IKJADM1" IKJADM1
SAY "IKJADM2" IKJADM2
SAY "IKJADM3" IKJADM3
```

IKJADM enthält die Anzahl der erzeugten Meldungen. IKJADM1 und folgende enthalten die Meldungen (Satzlänge 151 Byte).