Natural-Stack

Der Natural-Stack ist eine Art "Zwischenablage", in der Sie Natural-Kommandos, Benutzerkommandos und Daten für ein INPUT-Statement speichern können.

Dieses Kapitel behandelt folgende Themen:

- Verwendung des Natural-Stack
- Stack-Verarbeitung
- Daten im Stack ablegen
- Stack-Inhalt löschen

Verwendung des Natural-Stack

So können Sie häufig nacheinander ausgeführte Funktionen, wie beispielsweise eine Abfolge von Logon-Kommandos, die häufig in der gleichen Reihenfolge ausgeführt werden, speichern.

Der Stack ist mit einem Stapel vergleichbar: die Daten/Kommandos werden aufeinander "gestapelt" und können sowohl oben auf dem Stack als auch unten im Stack abgelegt werden. Die gespeicherten Daten/Kommandos können nur in der gestapelten Reihenfolge verarbeitet werden, und zwar von oben nach unten.

Mit der Systemvariable *DATA können Sie sich in einem Programm den Inhalt des Stack anzeigen lassen (weitere Informationen siehe *Systemvariablen*-Dokumentation).

Stack-Verarbeitung

Die Verarbeitung der im Stack gespeicherten Kommandos und Daten ist abhängig von der jeweils ausgeführten Funktion.

Falls ein Kommando einzugeben ist, d.h. falls als nächstes die NEXT-Zeile erscheinen müsste, sucht Natural den Stack von oben nach unten nach einem Kommando ab; wird ein Kommando gefunden, so wird die NEXT-Zeile unterdrückt, das Kommando gelesen und aus dem Stack gelöscht. Das Kommando wird ausgeführt, als wäre es von Hand in der NEXT-Zeile eingegeben worden.

Falls ein INPUT-Statement ausgeführt wird, das Eingabefelder enthält, sucht Natural, bevor der INPUT-Schirm angezeigt wird, den Stack nach Eingabedaten ab und übergibt diese automatisch an das INPUT-Statement (und zwar unter Delimiter-Mode-Logik). Natural überprüft, ob es sich um für das betreffende INPUT-Statement gültige Eingabedaten handelt; anschließend löscht es die Daten aus dem Stack. Siehe auch *INPUT-Daten aus dem Natural-Stack* in der Beschreibung des INPUT-Statements.

Falls ein INPUT-Statement mit Stack-Daten ausgeführt wird und dieses INPUT-Statement durch ein REINPUT-Statement nochmals ausgeführt wird, wird der INPUT-Schirm angezeigt, und zwar mit den gleichen Stack-Daten wie beim ersten Mal. Bei einem REINPUT-Statement werden keine weiteren Daten vom Stack gelesen.

Wird ein Natural-Programm normal beendet, werden die zuoberst gelagerten Daten im Stack soweit gelöscht, bis sich entweder oben im Stack wieder ein Kommando befindet oder der Stack ganz geleert ist. Wird ein Natural-Programm aufgrund eines Fehlers oder mit dem Terminalkommando %% abgebrochen, wird der gesamte Inhalt des Stacks gelöscht.

Daten im Stack ablegen

Es gibt folgende Möglichkeiten, Daten bzw. Kommandos im Stack abzulegen:

- STACK-Parameter
- STACK-Statement
- FETCH- und RUN-Statements

STACK-Parameter

Sie können den Natural-Profilparameter STACK benutzen, um Daten oder Kommandos im Stack abzulegen. Der STACK-Parameter, der in der Natural *Parameter-Referenz*-Dokumentation beschrieben ist, kann bei der Installation von Natural vom Natural-Administrator im Natural-Parametermodul gesetzt werden. Sie können den STACK-Parameter auch als dynamischen Parameter beim Aufruf von Natural angeben.

Werden Daten/Kommandos mit dem STACK-Parameter im Stack abgelegt, so müssen mehrere Kommandos mit einem Semikolon (;) voneinander getrennt werden. Einem Kommando, das innerhalb einer Reihe von Daten- bzw. Kommandoelementen übergeben wird, muss ein Semikolon vorangestellt werden.

Daten für mehrere INPUT-Statements müssen mit einem Doppelpunkt (:) voneinander getrennt werden. Einer Datenkette, die von einem weiteren INPUT-Statement gelesen werden soll, muss jeweils ein Doppelpunkt vorangestellt werden. Soll ein Kommando, das Parameter erfordert, im Stack abgelegt werden, werden das Kommando und die dazugehörigen Parameter nicht durch einen Doppelpunkt voneinander getrennt.

Doppelpunkt und Semikolon dürfen nicht in den für das INPUT-Statement bestimmten Daten selbst auftauchen, da sie als Trennzeichen interpretiert werden.

STACK-Statement

Innerhalb eines Natural-Programms können Sie das STACK-Statement verwenden, um Daten oder Kommandos im Stack abzulegen. Die in einem STACK-Statement angegebenen Datenelemente können nur für ein einziges INPUT-Statement verwendet werden; d.h. Sie müssen mehrere STACK-Statements verwenden, wenn Sie Daten für mehrere INPUT-Statements im Stack ablegen wollen.

Daten können entweder formatiert oder unformatiert im Stack abgelegt werden:

- Werden unformatierte Daten aus dem Stack gelesen, werden sie im Delimiter-Modus interpretiert, wobei die mit den Session-Parametern IA (Input Assign) und ID (Input Delimiter) festgelegten Zeichen als Input-Zuweisungszeichen bzw. -Trennzeichen verarbeitet werden.

- Formatiert im Stack gelagerte Daten werden nach Feldinhalten getrennt und Feld für Feld an die Eingabefelder des betreffenden INPUT-Statements übergeben. Falls die im Stack abzulegenden Daten Begrenzungs-, Steuer- oder DBCS-Zeichen enthalten, sollten sie, um eine unbeabsichtigte Interpretation dieser Zeichen zu vermeiden, formatiert im Stack abgelegt werden.

Eine ausführliche Beschreibung des Statements STACK finden Sie in der *Statements*-Dokumentation.

FETCH- und RUN-Statements

Werden bei der Ausführung eines FETCH- oder RUN-Statements Parameter an das aufgerufene Programm übergeben, so werden diese Parameter oben auf dem Stack abgelegt.

Stack-Inhalt löschen

Der Inhalt des Stacks kann mit dem Statement RELEASE gelöscht werden. Eine ausführliche Beschreibung des Statements finden Sie in der *Statements*-Dokumentation.

Anmerkung:
Wenn ein Natural-Programm mittels des Terminalkommandos %% oder mit einem Fehler beendet wird, wird der Stack vollständig gelöscht.