Natural for z/OS

Unicode- und Codepage-Unterstützung

Version 9.2.5

Oktober 2026

ADABAS & NATURAL

This document applies to Natural for z/OS Version 9.2.5 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1979-2026 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software GmbH product names are either trademarks or registered trademarks of Software GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Document ID: NATMF-NATUNICODE-925-20261013DE

Inhaltsverzeichnis

Vorwort ... v
1 Über diese Dokumentation ... 1
Dokumentationskonventionen ... 2
Online-Informationen und Support ... 2
Datenschutz ... 3
2 Einführung ... 5
Allgemeines zu Codepages und Unicode ... 6
Unicode- und Codepage-Unterstützung in Natural ... 7
3 Unicode- und Codepage-Unterstützung aktivieren ... 9
ICU for Adabas & Natural (ICS) ... 10
ICS-Modul SAGICU ... 10
Alternative ICS-Module zur Unterstützung von Architecture Levels ... 12
ICU Data Libraries ... 13
ICU-Datenelemente ... 13
Unicode- und Codepage-Support für Adabas ... 15
Umsetzungstabellen (Translation Tables) ... 15
Unterstützung von Multi-Byte-Codepages ... 16
4 ICS 322 ... 17
Umfang und Handhabung der Daten ... 18
Schlüsselwort-Subparameter STEPLIB im Profilparameter CFICU ... 18
Validierung des STEPLIB-Subparameterwerts im Profilparameter CFICU ... 19
ICS32 in einer Natural-Subtask verwenden (benötigt APF-autorisierte ICS-Ladebibliothek) ... 19
5 Unicode-/Codepage-Umgebung konfigurieren und verwalten ... 21
ICU Data Library kundenspezifisch einrichten ... 22
Natural-Profilparameter und Parameter-Makros ... 26
Information zur Zeichencodierung ... 31
6 Entwicklungsumgebung ... 33
Entwicklungsumgebung für Anwendungen ... 34
Kundenspezifische Anpassung Ihrer Umgebung ... 35
Editoren in der SPoD-Umgebung ... 35
Codepage-Unterstützung bei Editoren, Systemkommandos und Utilities ... 36
Codepage-Unterstützung bei Natural-Quellcode-Objekten ... 39
7 Unicode- und Codepage-Unterstützung in der Natural-Programmiersprache ... 43
Natural-Datenformat U für Unicode-basierende Daten ... 44
Natural-Statements ... 45
Logische Bedingungen ... 50
Systemvariablen ... 50
Große und dynamische Variablen ... 51
Session-Parameter ... 51
Beispiel-Programme ... 54
8 Behandlung von Unicode-Ein-/Ausgaben in Natural-Anwendungen ... 55
Unicode-Daten anzeigen und eingeben ... 56

Natural Web I/O Interface Client 57
9 Unterstützung bidirektionaler Sprachen 61
Allgemeines zu bidirektionalen Sprachen 62
Zeichenlaufrichtung im Bildschirm 62
Zeichenlaufrichtung im Feld 63
Arabische Formgebung 64
10 Unicode-Datenspeicherung 67
Zugriff auf Unicode-Daten und Parameter 68
Datenbankmanagementsystem-Schnittstellen 68
Arbeitsdateien und Druckdateien 69
11 Migration existierender Anwendungen 71
Einfluss von Unicode auf existierende Anwendungen 72
Existierende Objekte migrieren 72
Existierende Anwendungen auf Unicode-Unterstützung erweitern 74
Migration von Natural Remote Procedure Calls (RPC) 74
12 Häufig gestellte Fragen 75
Warum erhalte ich beim Start den Fehler 'Invalid code page specified'? 76
Was ist die Standard-Codepage? 76
Welche Standard-Codepage wird benutzt? 76
Wie kann ich alle relevanten Natural-Codepage-Einstellungen anzeigen? 76
Wie kann ich UTF-8-Zeichencodierung mit Natural-Code behandeln? 76
Warum werden manche Zeichen nicht korrekt angezeigt? 77
Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt bearbeiten möchte? 77
Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt speichern möchte? 77
Wie kann ich die Zeichencodierung eines Natural-Quellcodes herausfinden? 78
Wie kann ich die Zeichencodierung eines Natural-Quellcodes ändern? 78
Welches Ersatzzeichen wird verwendet, wenn ein Zeichen nicht konvertiert werden kann? 78
Kann ich Natural-Quellcodes mit früheren Natural-Versionen benutzen, die noch keine Codepage-Unterstützung bieten? 78
Stichwortverzeichnis 79

Vorwort

Diese Dokumentation beschreibt, wie Unicode, Codepages und bidirektionale Sprachen von Natural auf Großrechner-Plattformen unterstützt werden.

Diese Dokumentation ist in die folgenden Kapitel untergliedert:

Einführung	Allgemeine Informationen zu Codepages und zum Unicode-Standard. Hinweise dazu, auf welche Weise Unicode und Codepages in Natural unterstützt werden.
Unicode- und Codepage-Unterstützung aktivieren	Informationen zu ICU for Adabas & Natural (ICS), ICU Data Libraries, Unterstützung von Codepages und zu Transaktionstabellen.
ICS 322	Informationen zur Version ICS 322, die auf ICU 66.1 und Unicode-Version 13.0 basiert.
Unicode-/Codepage-Umgebung konfigurieren und verwalten	Informationen zu Natural-Profilparametern, die zum Aktivieren und Einstellen der Unicode- und Codepage-Unterstützung dienen, und zur Zeichenkodierung von Codepage-Daten.
Entwicklungsumgebung	Wie Sie Ihre Umgebung kundenspezifisch anpassen und wie Unicode von den Natural-Editoren behandelt wird. Informationen zur Codepage-Unterstützung bei Natural-Editoren, Systemkommandos und Natural-Quellcode-Objekten.
Unicode- und Codepage-Unterstützung in der Natural-Programmiersprache	Informationen zum Format U und zu Statements, logischen Bedingungen, Systemvariablen, großen und dynamischen Variablen sowie zu Session-Parametern zur Unicode- und Codepage-Unterstützung.
Unicode-Eingabe-/Ausgabebehandlung in Natural-Anwendungen	Wie Sie Unicode-Daten anzeigen und eingeben. Informationen zum Natural Web I/O Interface Client, der in Natural-SPoD- und Laufzeit-Umgebungen benutzt wird.
Unterstützung bidirektionaler Sprachen	Wie bidirektionale Sprachen in Natural unterstützt werden.
Unicode-Datenspeicherung	Informationen zum Datenbankzugriff und zu den Arbeitsdateitypen und Druckdateien, die Unicode- und Codepage-Unterstützung bieten.
Migration existierender Anwendungen	Auswirkungen von Unicode auf existierende Anwendungen. Wie Sie existierende Objekte migrieren, Unicode-Unterstützung bei existierenden Anwendungen verfügbar machen und Natural Remote Procedure Calls (RPC) migrieren.
Häufig gestellte Fragen	Antworten auf häufig gestellte Fragen.

1 Über diese Dokumentation

- Dokumentationskonventionen 2
- Online-Informationen und Support 2
- Datenschutz 3

Dokumentationskonventionen

Konvention	Beschreibung
Fettschrift	>Kennzeichnet Elemente auf einem Bildschirm.
`Nichtproportionale Schrift`	Kennzeichnet Namen und Orte von Diensten im Format *`Ordner.Unterordner.Dienst`*, Programmierschnittstellen (APIs), Namen von Klassen, Methoden und Properties in Java.
`Kursivschrift`	Kennzeichnet: Variablen, für die Sie situations- oder umgebungsspezifische Werte angeben müssen. Neue Begriffe, wenn sie erstmals im Text auftreten. Verweise auf andere Dokumentationsquellen.
`Nichtproportionale Schrift`	Kennzeichnet: Text, den Sie eingeben müssen. Meldungen, die vom System angezeigt werden. Programmcode.
{ }	Zeigt eine Reihe von Auswahlmöglichkeiten an, von denen Sie eine auswählen müssen. Geben Sie nur die innerhalb der geschweiften Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole { } ein.
`|`	Trennt zwei sich gegenseitig ausschließende Auswahlmöglichkeiten in einer Syntaxzeile voneinander ab. Geben Sie eine der Auswahlmöglichkeiten ein. Geben Sie nicht das Symbol `|` ein.
[ ]	Zeigt eine oder mehrere Optionen an. Geben Sie nur die innerhalb der eckigen Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole [ ] ein.
...	Zeigt an, dass Sie mehrere Auswahlmöglichkeiten desselben Typs eingeben können. Geben Sie nur die Informationen ein. Geben Sie nicht die drei Auslassungspunkte (...) ein.

Online-Informationen und Support

Produktdokumentation

Sie finden die Produktdokumentation auf unserer Dokumentationswebsite unter **https://documentation.softwareag.com**.

Zusätzlich können Sie auch über **https://www.softwareag.cloud** auf die Dokumentation für die Cloud-Produkte zugreifen. Navigieren Sie zum gewünschten Produkt und gehen Sie dann, je nach Produkt, zu „Developer Center“, „User Center“ oder „Documentation“.

Produktschulungen

Sie finden hilfreiches Produktschulungsmaterial auf unserem Lernportal unter **https://knowledge.softwareag.com**.

Tech Community

Auf der Website unserer Tech Community unter **https://techcommunity.softwareag.com** können Sie mit Experten der Software AG zusammenarbeiten. Von hier aus können Sie zum Beispiel:

- Unsere umfangreiche Wissensdatenbank durchsuchen.
- In unseren Diskussionsforen Fragen stellen und Antworten finden.
- Die neuesten Nachrichten und Ankündigungen der Software AG lesen.
- Unsere Communities erkunden.
- Unsere öffentlichen Repositories auf GitHub and Docker unter **https://github.com/softwareag** und **https://hub.docker.com/publishers/softwareag** besuchen und weitere Ressourcen der Software AG entdecken.

Produktsupport

Support für die Produkte der Software AG steht lizenzierten Kunden über unser Empower-Portal unter **https://empower.softwareag.com** zur Verfügung. Für viele Dienstleistungen auf diesem Portal benötigen Sie ein Konto. Wenn Sie noch keines haben, dann können Sie es unter **https://empower.softwareag.com/register** beantragen. Sobald Sie ein Konto haben, können Sie zum Beispiel:

- Produkte, Aktualisierungen und Programmkorrekturen herunterladen.
- Das Knowledge Center nach technischen Informationen und Tipps durchsuchen.
- Frühwarnungen und kritische Alarme abonnieren.
- Supportfälle öffnen und aktualisieren.
- Anfragen für neue Produktmerkmale einreichen.

Datenschutz

Die Produkte der Software AG stellen Funktionen zur Verarbeitung von personenbezogenen Daten gemäß der Datenschutz-Grundverordnung (DSGVO) der Europäischen Union zur Verfügung. Gegebenenfalls sind in der betreffenden Systemverwaltungsdokumentation entsprechende Schritte dokumentiert.

2 Einführung

- Allgemeines zu Codepages und Unicode 6
- Unicode- und Codepage-Unterstützung in Natural 7

Allgemeines zu Codepages und Unicode

Eine herkömmliche Zeichensatztabelle (im Folgenden „Codepage“ genannt) besteht aus einer Liste mit ausgewählten Zeichencodes, die in einer festgelegten Reihenfolge angeordnet sind und bestimmte Sprachen oder Gruppen von Sprachen unterstützen, die gemeinsame Schriftzeichen haben. Eine Codepage kann maximal 256 Zeichencodes haben. Bei Zeichensätzen, die mehr als 256 Zeichen enthalten (z.B. Chinesisch oder Japanisch) kommt ein Double-Byte Character Set (DBCS) zur Anwendung: DBCS-Codepages sind faktisch mehrere Bytes umfassende Zeichenkodierungen („Encodings"“), eine Mischung aus 1-Byte- und 2-Byte-Codepunkten („Codepoints“).

Codepages haben den inhärenten Nachteil, dass sie nicht verwendet werden können, um verschiedene Sprachen im selben Datenstrom zu speichern. Unicode wurde entworfen, um diese Einschränkung zu beseitigen. In Unicode wird eine Standard-Zeichencodierung für alle Zeichensätze zur Verfügung gestellt, die unabhängig von der Plattform, dem Programm oder der Sprache ist, die für den Datenzugriff benutzt wird. In Unicode steht für jedes Zeichen eine eindeutige Zahl zur Verfügung.

Jedem durch den Unicode-Standard definierten Code-Element wird ein Zahlenwert im Coderaum zugewiesen. Dieser Zahlenwert wird als „Codepoint“ bezeichnet. Bei Bezugnahme im Text wird der Codepoint in hexadezimaler Form mit einem vorangestellten **U** aufgeführt. Beispiel: Der Codepoint "U+0041" ist die Hexadezimalzahl "0041" (äquivalent zur Dezimalzahl "65"). Er repräsentiert das große "A" im Unicode-Standard und trägt den Namen „LATIN CAPITAL LETTER A“.

Der Unicode-Standard definiert drei Zeichencodierungsformen, mit denen die gleichen Daten in einem Byte-, Wort- oder Doppelwort-orientierten Format übertragen werden können. Eine Codeeinheit („Code Unit“) ist die geringste Bit-Kombination, mit der ein Zeichen in einer spezifischen Zeichenkodierung dargestellt werden kann. Der Unicode-Standard verwendet 8-Bit-Codeeinheiten in der Encoding Form (Abbildung der Codepoints auf Codeeinheiten) UTF-8, 16-Bit-Codeeinheiten in der Encoding Form UTF-16 und 32-Bit-Codeeinheiten in der Encoding Form UTF-32. Alle drei Zeichencodierungsformen kodieren *denselben* gemeinsamen Zeichenvorrat und können zügig und ohne Datenverlust ineinander umgewandelt werden.

Im Zusammenhang mit Natural haben wir es mit zwei dieser Zeichencodierungsformen zu tun: UTF-16 und UTF-8. Natural benutzt UTF-16 für das Kodieren von Unicode-Zeichenketten zur Laufzeit und UTF-8 für das Kodieren von Unicode-Daten in Dateien. UTF-16 ist eine Endian-abhängige 2-Byte-Kodierung. Das verwendete Endian-Format ist plattformabhängig. UTF-8 ist eine Zeichenkodierungen mit variabler Länge.

Eine vollständige Beschreibung des Unicodes finden Sie auf der Website des Unicode-Konsortiums unter ***http://www.unicode.org/***.

Anmerkung: Informationen zu Unicode-Codepoints erhalten Sie, wenn Sie die Natural Utility SYSCP benutzen.

Unicode- und Codepage-Unterstützung in Natural

Zur Unterstützung von Unicode werden das Natural-Datenformat U und spezifische Statements, Parameter und Systemvariablen benutzt. Diese werden in den folgenden Kapiteln ausführlich behandelt.

Die meisten bereits existierenden Daten liegen im Codepage-Format vor. Beim Umwandeln dieser Daten in Unicode muss die korrekte Codepage verwendet werden. Natural bietet die Möglichkeit, die korrekte Codepage auf mehreren Ebenen zu definieren:

- **System-Codepage**
  Die System-Codepage wird benutzt, wenn in Natural keine Standard-Codepage definiert ist.

  Wenn keine Codepage definiert wird (`CP=OFF`, siehe CP - Name der Standard-Codepage), ist keine Standard-Codepage definiert. Die Parametereinstellung `CP=AUTO` bewirkt, dass die Natural-Session an die Codepage des aktuellen Eingabe-/Ausgabegeräts angepasst wird.
- **Standard-Codepage**
  Die Standard-Codepage wird benutzt, wenn der Natural-Profilparameter `CP` auf einen Wert ungleich `OFF` gesetzt ist. Diese Einstellung hat Vorrang vor der Codepage des Betriebssystems.
- **Objekt-Codepage**
  Die Objekt-Codepage, die zum Beispiel für ein Quellcode-Objekt definiert wird, hat bei diesem Objekt Vorrang vor der Standard-Codepage.

Bei Verwendung von Unicode-Zeichenketten und Codepage-Zeichenketten innerhalb einer Anwendung, führt Natural eine implizite Umwandlung durch, wo immer dies nötig ist (beispielsweise beim Übertragen (`MOVE`) oder Vergleichen von Daten). Explizite Umwandlungen können mit dem Natural-Statement `MOVE ENCODED` durchgeführt werden.

In den meisten Fälle werden existierende Anwendungen, die keine Unicode-Unterstützung benötigen, unverändert laufen. Änderungen können nötig sein, wenn die existierenden Quellcode-Objekte in unterschiedlichen Codepages kodiert sind. Weitere Informationen siehe ***Migration existierender Anwendungen***.

Es ist nicht möglich, eine existierende Anwendung laufen zu lassen und Unicode-Daten zu unterstützen, ohne Änderungen an der Anwendung vorzunehmen. In der Anwendung muss das Natural-Datenformat U eingeführt werden und wahrscheinlich reicht es nicht aus, einfach die vorhandenen Definitionen im Format A durch Definitionen im Format U zu ersetzen. Es ist unumgänglich, jeglichen Code anzupassen, in dem ein bestimmtes Speicher-Layout für Zeichenketten erwartet wird (z.B. bei einer Redefinition von alphanumerischem in numerisches Format per `REDEFINE`-Statement).

Die Verwendung von Unicode-Zeichen ist unzulässig innerhalb von Variablennamen, Objektnamen und Library-Namen.

Daten auf Unicode-Basis werden von Adabas und DB2 unterstützt.

Natural verwendet die International Components for Unicode (ICU) Library für die Sortierung („Collation“) und Umwandlung von Unicode-Zeichen. Weitere Informationen siehe ***http://user-guide.icu-project.org/***. Siehe auch ***ICU for Adabas & Natural (ICS)*** weiter unten in dieser Dokumentation.

3 Unicode- und Codepage-Unterstützung aktivieren

- ICU for Adabas & Natural (ICS) 10
- ICS-Modul SAGICU 10
- Alternative ICS-Module zur Unterstützung von Architecture Levels 12
- ICU Data Libraries 13
- ICU-Datenelemente 13
- Unicode- und Codepage-Support für Adabas 15
- Umsetzungstabellen (Translation Tables) 15
- Unterstützung von Multi-Byte-Codepages 16

ICU for Adabas & Natural (ICS)

Für die Konvertierung von Codepages und der Unterstützung von Unicode kommt Funktionalität zur Anwendung, die im Rahmen der ICU for Adabas & Natural (ICS) zur Verfügung steht. Damit Natural zur Unicode- und Codepage-Unterstützung in der Lage ist, müssen die im ICS gelieferten Komponenten installiert werden: Das **ICS-Modul** `SAGICU` oder ein **alternatives ICS-Modul** (nur bei z/OS) und **ICU Data Libraries**.

Anmerkungen:

1. Zur Ausführung von Anwendungen ohne Unicode- und Codepage-Unterstützung braucht keine ICS-Komponente installiert zu werden, das heisst, wenn die Profilparameter `CFICU` und `CP` auf `OFF` gesetzt sind.
2. Informationen zur aktuell benutzten ICU-Version und Unicode Specification werden im Hauptmenü der Utility `SYSCP` angezeigt. Siehe *SYSCP Utility aufrufen und beenden* in der *Debugger und Dienstprogramme (Utilities)*-Dokumentation.

ICS-Modul SAGICU

Damit Natural zur Unicode- und Codepage-Unterstützung in der Lage ist, muss bei der Installation von Natural eine ICU Data Library verlinkt und geladen werden. Weitere Informationen siehe *Installing International Components for Unicode for Software AG* for z/OS (siehe ***ICS 322***).

Das ICS-Modul `SAGICU` ist dafür vorgesehen, unabhängig von Lokalisierungsdaten benutzt zu werden. Es enthält keine statisch gelinkten Codepages und Locales. Ein Dataset, der die gesamten, als Data Items modularisierten ICU-Lokalisierungsdaten enthält, wird mit ICS 322 ausgeliefert. Der Dataset-Name kann mit dem Schlüsselwort-Subparameter `STEPLIB` des Profilparameters `CFICU` oder statisch in der JCL als eine Natural Steplib angegeben werden.

- Collation Services

- Codepages und Locales

Collation Services

Eine weitere Besonderheit dieses Moduls sind die Collation Services. Diese werden verwendet, um Unicode-Zeichenketten miteinander zu vergleichen. Dabei wird berücksichtigt, dass die alphabetische Sortierfolge von Sprache zu Sprache unterschiedlich ist. Die Sprachen und Schreibsysteme dieser Welt und die jeweils verwendeten unterschiedlichen Einsortierungsregeln zu akkommodieren stellt eine große Herausforderung dar. Die ICU Collation Services stellen jedoch ausgezeichnete Mittel zur Verfügung, mit denen Zeichenketten unter Berücksichtung von Gebietsschemaparametern (Locales) verglichen werden können.

Beispiele: Beim deutschen Locale wird das Zeichen "Ä" zwischen "A" und "B" einsortiert, beim schwedischen Locale nach dem "Z". Im Litauischen wird das Zeichen "y" zwischen "i" und "k" einsortiert.

Die ICU-Implementierung der Collation Services ist konform mit dem Unicode Collation Algorithm und mit ISO 14651. Die Algorithmen wurden von Experten auf dem Gebiet der mehrsprachen Sortierung entworfen und überprüft und sind deshalb stabil und umfassend.

Codepages und Locales

Statisch gelinkte Collation Data zur Sortierung (ein Satz Codepages und Locale IDs) werden bei ICS 322 nicht unterstützt.

ICS 322 verwendet alle ICU-Lokalisierungsdaten.

Das ICS-Modul `SAGICU` stellt folgende Codepages und Locales zur Verfügung:

Codepages	Locales
IBM037	de_DE
IBM273	en_US
IBM1025	es_ES
IBM1026	fr_FR
IBM1047	sv_SE
IBM1097	
IBM01140	
IBM01141	
IBM01145	
IBM01146	
IBM01147	
US (alias for IBM01140)	
DE (alias for IBM01141)	
ES (alias for IBM01145)	
EN (alias for IBM01146)	
FR (alias for IBM01147)	
IBM-37_P100-1995,SWAPLFNL	

Codepages	Locales
IBM-1047_P100-1995,SWAPLFNL IBM-1140_P100-1997,SWAPLFNL EBCDIC-XML-US IBM-290 (Japanese code page SBCS) IBM-930 (Japanese code page SBCS/DBCS) IBM-939 (Japanese code page SBCS/DBCS) IBM-1390 (Japanese code page SBCS/DBCS) IBM-1399 (Japanese code page SBCS/DBCS) IBM-932 (Japanese code page ASCII MBCS) IBM-942 (Japanese code page ASCII MBCS) IBM-943 (Japanese code page ASCII MBCS) EUC-JP (Japanese code page ASCII MBCS) IBM-420 (RTL code page) IBM-424 (RTL code page) IBM-916 (RTL code page)	

Alternative ICS-Module zur Unterstützung von Architecture Levels

Wenn Ihr Natural-System auf z/OS mit einem IBM-Prozessor mit Architecture Level 9 oder höher läuft, können Sie das ICS-Modul `SAGICU` durch `SAGICUA9` ersetzen. `SAGICUA9` ist so aufgebaut, dass es erweiterte Maschineninstruktionen nutzt, die mit der ESA/390 und z/Architecture von IBM eingeführt worden sind. Sie können das Systemkommando `TECH` benutzen (siehe *Systemkommandos*-Dokumentation), um festzustellen, welcher Architecture Level auf Ihrer derzeitigen Maschine unterstützt wird.

`SAGICUA9` bewirkt eine Performance-Verbesserung insbesondere bei der Ausführung von Natural-Statements, die Unicode-Variablen oder Instruktionen zur Codepage-Zeichencodierung benutzen (z.B. `MOVE ENCODED`). Weitere Informationen siehe themenspezifische IBM-Dokumentation (*z/Architecture, Principles of Operation*).

Vorsicht: Ein Operation Exception Error (Abend Code S0C1) kann auftreten, wenn das ICS-Modul `SAGICUA9` verwendet wird, aber der Architecture Level der zugrunde liegenden Maschine niedriger als 9 ist.

ICU Data Libraries

Wenn Sie Natural für die Unicode- und Codepage-Unterstützung aktivieren wollen, müssen Sie bei der Installation von Natural eine ICU Data Library verlinken und laden, siehe *Installing International Components for Unicode for Software AG* für z/OS.

ICU Data Libraries werden mit den folgenden ICU Data Modules geliefert. Dabei steht `nn` für die aktuelle Version des Moduls entsprechend der Ankündigung in der aktuellen Natural-*Freigabemitteilung (Release Notes)*.

Datenmodul	Beschreibung
`ICSDTnnE`	Enthält die gebräuchlichsten Codepages und Locales. Die Codepages sind schon in `NATCONFG` deklariert.
`ICSDTnnJ`	Wie bei `ICSDTnnE`, jedoch erweitert um japanische Codepages. `ICSDTnnJ` ist schon mit dem ICS-Modul `SAGICU` (oder einem **alternativen ICS-Modul** auf z/OS) verlinkt. Es enthält die oben genannten Codepages und Locales.
`ICSDTnnX`	Enthält alle möglichen Converters und Locales, die von den zurzeit unterstützten ICU-Versionen angeboten werden. Es unterstützt ca. 230 verschiedene Codepages (überwiegend EBCDIC-Codepages) und 238 Locales. Deshalb ist das Modul sehr groß. `ICSDTnnX` unterstützt alle Codepages und Locale IDs, die von der zurzeit unterstützten ICU-Version unterstützt werden (siehe ***http://demo.icu-project.org/icu-bin/convexp***).

Sie haben die Möglichkeit Ihrer eigene ICU Data Library zu erstellen, die exakt Ihren Erfordernissen entspricht (siehe ***ICU Data Library kundenspezifisch einrichten***).

ICU-Datenelemente

Die von Natural unterstützten ICU-Datenelemente (Data Items) beinhalten Converters und Collators. Beispiel: Ein Converter wird benutzt, wenn ein `MOVE ENCODED`-Statement ausgeführt wird, und ein Collator, wenn Zeichenketten in einem `IF`-Statement verglichen werden.

Ein ICU-Datenelement ist entweder statisch mit einer ICU Data Library verlinkt oder es wird auf Anforderung während der Natural-Session dynamisch geladen.

ICU-Datenelemente werden als ladbare Module in dem für die Installation von Natural gelieferten ICU-Dataset mitgeliefert und müssen über die Natural-Steplib-Kette zugänglich sein.

Wird ein ICU-Datenelement zum ersten Mal aufgerufen, versucht ICS, es aus der verlinkten oder geladenen ICU Data Library zu öffnen. Falls kein ICU-Datenelement mit einer Library verbunden ist, versucht ICS, das Datenelement dynamisch aus dem ICS-Dataset zu laden.

Folgende Themen werden behandelt:

- Namenskonventionen für Datenelementmodule
- Dynamisch geladene einzelne ICU-Datenelemente

Namenskonventionen für Datenelementmodule

Die Länge des Namens eines Datenelementmoduls im ICS-Dataset ist auf acht Zeichen beschränkt. Wie in der folgenden Tabelle aufgeführt, hat der Name folgende Bestandteile:

- Ein Präfix (`I`),
- eine zweistellige ICU-Version (*xx*),
- eine logische Gruppenkennung (`C`, `B`, `S`, `L`, `M` oder `D`) und
- eine vierstellige Folgenummer (*nnnn*).

Modulname	Inhalt
I*xx*C*nnnn*	Zeichensatz-Abbildungstabellen (Converter-Module)
I*xx*B*nnnn*	Umbruch-Iteratoren
I*xx*S*nnnn*	Collators (Collation Services)
I*xx*L*nnnn*	Lokalisierung (Formatierung, Anzeigenamen und sonstige lokalisierten Daten)
I*xx*M*nnnn*	Verschiedene Daten (regelbasierte Zahlenformate und Transliteratoren)
I*xx*D*nnnn*	Basisdaten

Beispiel:

`I58C0074` ist der Name eines Converters für ICU Version 58.2 und Codepage ibm-1148_P100-1997.

In einem `MOVE ENCODED`-Statement erwartet Natural jedoch den Langnamen der Codepage, die dem Datenelementmodul entspricht. Es kann jeder gültige Aliasname der Codepage verwendet werden. Der Name der Codepage wird automatisch auf den achtstelligen Kurznamen abgebildet, wenn das Datenelementmodul geladen wird.

Weitere Informationen siehe relevante ICU Website.

Dynamisch geladene einzelne ICU-Datenelemente

Die Verwendung von dynamisch geladenen individuellen Datenelementmodulen ermöglicht eine hohe Flexibilität. Die Daten werden bei Bedarf geladen und unterstützen alle Codepages. Ein Dataset mit allen ICU-Lokalisierungsdaten, modularisiert in einzelne Datenelemente, ist Teil der ICS 322-Lieferung.

Ein einzelnes Datenelementmodul wird beim erstmaligen Zugriff (z.B. durch ein `MOVE ENCODED`-Statement) geladen und ist für die künftige Verwendung sofort verfügbar, ohne dass es neu geladen werden muss. Nur die bereits verwendeten Codepages werden im Speicher gehalten,

jedoch keine statisch verlinkten Daten oder eine separate Data Library, wie es bei früheren ICS-Versionen der Fall war.

Unicode- und Codepage-Support für Adabas

Wenn eine Natural-Session für die Codepage- und Unicode-Unterstützung aktiviert ist, sollten Sie sich vergewissern, dass Naturals Adabas-Benutzer-Session ebenfalls die entsprechende Zeichencodierung (Encoding) für den Zugriff auf Adabas-Daten verwendet.

Da Adabas für die Konvertierung die Entire Conversion Services (ECS) benutzt, muss der ECS-Name in dem zugehörigen `NTCPAGE`-Eintrag im Modul `NATCONFG` angegeben sein. Um sicherzustellen, dass Naturals Adabas-Benutzer-Session die richtige Codepage benutzt, müssen Sie die Option `ACODE` und/oder `WCODE` im `OPRB`-Parameter für die benutzten Datenbanken angeben.

Weitere Informationen zur Adabas-Unicode- und Codepage-Unterstützung siehe Adabas-Dokumentation für Großrechner.

Umsetzungstabellen (Translation Tables)

Natural benutzt zahlreiche Tabellen zur Umsetzung von Zeichen und zur Festlegung der Eingeschaften eines Zeichens. Der Inhalt der Tabellen kann während des Starts einer Natural-Session über die entsprechenden Profilparameter geändert werden: `TAB`, `UTAB1`, `UTAB2` und `SCTAB`.

Wenn Natural mit Codepage-Unterstüzung läuft (d.h. wenn der Profilparameter `CP` auf einen Wert ungleich `OFF` gesetzt ist), können die Tabelleninhalte nicht durch den Benutzer geändert werden. In diesem Fall erscheint beim Natural-Start die folgende Meldung, um den Benutzer in Kenntnis zu setzen, dass die oben erwähnten Session-Parameter nicht berücksichtigt werden:

`Character translation parameter` *`table-name`* `ignored due to CFICU=ON.`

(Zeichenumsetzungsparameter *`table-name`* wird ignoriert wegen CFICU=ON.)

Natural passt die Tabellen automatisch entsprechend der für die Natural-Session benutzten Codepage (Wert der Systemvariablen `*CODEPAGE`) an. Siehe auch *Umsetzungstabellen* in der *Operations*-Dokumentation.

Unterstützung von Multi-Byte-Codepages

Natural unterstützt Multi-Byte-Codepages (MBCS), z.B. BM-939, bei der es sich um eine japanische Codepage handelt, die auf EBCDIC und DBCS basiert. Die Auswahl einer Multi-Byte-Codepage kann mit dem Natural-Profilparameter `CP` erfolgen, indem die Einstellung `AUTO=AUTO` (falls unterstützt) gewählt oder der Name der Codepage angegeben wird. Wenn Natural mit einer Multi-Byte-Codepage läuft, benutzt es interne Ein-/Ausgabepuffer, die auf Unicode basieren. Das bedeutet, dass alle Daten, die von einem Ein-/Ausgabe-Statement in die internen Ein-/Ausgabepuffer geschrieben werden, in Unicode konvertiert werden. Aufrund der von Unicode und Multi-Byte-Codepages gestellten Anforderungen sind die Puffer größer als bei herkömmlichen Ein- und Ausgaben, weil Unicode-Zeichen doppelt so viel Platz wie EBCDIC-Zeichen benötigen und weil verbesserte Attribute zur Beschreibung eines Feldes benötigt werden.

Im Falle von Single-Byte Codepages (SBCS), z.B. IBM-1140, werden noch die herkömmlichen, auf EBCDIC basierenden Ein- und Ausgaben zum Aufbewahren von Ressourcen benutzt.

4 ICS 322

- Umfang und Handhabung der Daten 18
- Schlüsselwort-Subparameter STEPLIB im Profilparameter CFICU 18
- Validierung des STEPLIB-Subparameterwerts im Profilparameter CFICU 19
- ICS32 in einer Natural-Subtask verwenden (benötigt APF-autorisierte ICS-Ladebibliothek) 19

Folgende Themen werden behandelt:

Umfang und Handhabung der Daten

In ICS 322 werden alle verfügbaren Daten zur kundenspezifischen Anpassung in modularisierter Form als so genannte **Datenelemente** (Data Items) zur Verfügung gestellt. Der Dataset, der die Datenelemente enthält, ist ebenso wie die ICS 322-Lademodule `SAGICU` und `SAGICUA9` im Lieferumfang enthalten.

Ein Datenelement (Collator, Converter) befindet sich im Dataset auf der Platte und wird bei Bedarf in den Speicher geladen (z.B. durch ein `MOVE ENCODED`-Statement). Sobald es geladen wird, ist es für die künftige Verwendung sofort verfügbar, ohne dass es neu geladen werden muss.

Dadurch kann die Größe der ICS 322-Lademodule `SAGICU` und `SAGICUA9` *minimal* gehalten werden.

Mit der Utility `SYSCP` können Sie sich alle geladenen Datenelemente auflisten lassen (siehe `SYSCP`-Funktion Geladene Codepages - Funktion: Loaded Code Pages in der *Debugger und Dienstprogramme (Utilities)*-Dokumentation).

Schlüsselwort-Subparameter STEPLIB im Profilparameter CFICU

Zur Verfügung gestellte Datendateien werden ab ICS 322 nicht mehr unterstützt. Die ICU-Lokalisierungsdaten werden nur dynamisch aus einem Dataset geladen, der die Datenelemente (Collators, Converters usw.) enthält.

Der Name des Dataset kann

- *statisch* angegeben werden als eine Natural-Steplib in der JCL.
- *dynamisch* zugeordnet werden mit dem Schlüsselwort-Subparameter `STEPLIB` des Profilparameters `CFICU`.

Um nach Datenelementen zu suchen, macht ICS von beiden Zuordnungsmethoden Gebrauch: zunächst bei dem im Schlüsselwort-Subparameter `STEPLIB` angegebenen Dataset (falls gegeben) und dann bei den statisch angegebenen Steplibs in der JCL.

Diese dynamische Vorgehensweise ermöglicht eine hohe Flexibilität. Die JCL braucht nicht geändert werden. Um Natural laufen zu lassen, muss nur in der Session ein Subparameter hinzugefügt werden.

Der mit dem Schlüsselwort-Subparameter `STEPLIB` des Profilparameters `CFICU` angegebene Dataset wird dynamisch nur einmal zugeordnet durch die erste Natural-Session in einem gegebenen TP-System unter der DD Karte ICS*xxx*DD (dabei steht *xxx* für die ICS-Version) und wird danach in allen Natural-Sessions benutzt.

Beispiel: `CFICU=(STEPLIB='I322ITEMS.LOAD')`

Validierung des STEPLIB-Subparameterwerts im Profilparameter CFICU

Die erste Natural-Session in einem gegebenen TP-System versucht, den Wert des Schlüsselwort-Subparameters `STEPLIB` im Profilparameter `CFICU` auf Gültigkeit zu prüfen. Verläuft die Gültigkeitsprüfung erfolgreich, wird ICS initialisiert. In allen anschließenden Natural-Sessions in dem TP-System wird der `STEPLIB`-Subparameter *nicht beachtet* und der in der ersten Session dynamisch zugeordnete Dataset verwendet. Für den mit dem `STEPLIB`-Subparameter angegebenen Dataset-Namen gelten folgende Validierungsregeln:

- Der Dataset muss existieren.
- Der Name des Dataset muss mit Hochkommas (`' '`) umschlossen sein, siehe Beispiel weiter oben.
- Der Name des Dataset muss den **z/OS-Namenskonventionen** entsprechen:
  - Maximale Länge: 44 Zeichen.
  - Darf keine Sonderzeichen enthalten.
  - Darf kein High Level Qualifier sein, d.h., er muss mindestens einen Punkt (`.`) enthalten.

War die Gültigkeitsprüfung nicht erfolgreich, wird der folgende Natural-Fehler ausgegeben:

```
NAT3414 STEPLIB DSN <data set name> cannot be loaded
```

Die nächste Natural-Session wird dann versuchen, den `STEPLIB`-Subparameter zu validieren und ICS neu zu initialisieren. Dieser Vorgang wird iterativ so lange fortgesetzt, bis eine erfolgreiche Initialisierung erreicht worden ist. Alle Sessions nach einer erfolgreichen ICS-Initialisierung werden den `STEPLIB`-Subparameter nicht beachten. Sie benutzen die bereits zugeordnete Ressource.

ICS32 in einer Natural-Subtask verwenden (benötigt APF-autorisierte ICS-Ladebibliothek)

Bei der aktuellen Implementierung des Entire System Server (NPR) muss die Ladebibliothek PRD.ICS*nnn*.MVSLOAD immer APF-autorisiert sein. Dies ist unabhängig davon, ob sie Teil der Steplib-Verkettung ist oder mit dem `STEPLIB`-Subparameter des Profilparameters `CFICU` angegeben wird.

Für die Verwendung von ICS31 in einer Natural-Subtask gibt es folgende Szenarien:

- Wenn mehrere Natural-Sitzungen/-Subtasks von einem Entire System Server (NPR)-Knoten aus laufen.

- Wenn Entire Operations (NOP), Entire Output Management (NOM), EOR automatisch von einem ESS/NPR-Knoten ausgeführt werden sollen.
- Wenn ein NDV-Server ausgeführt wird.

5 Unicode-/Codepage-Umgebung konfigurieren und verwalten

- ICU Data Library kundenspezifisch einrichten 22
- Natural-Profilparameter und Parameter-Makros 26
- Information zur Zeichencodierung 31

Notation *vr*:

In diesem Dokument stellt die Notation *vr* die aus zwei Zeichen bestehende ICU-Versionsnummer dar.

ICU Data Library kundenspezifisch einrichten

In ICU werden verschiedenste Datentabellen verwendet, um vielfältige Dienste zur Verfügung zu stellen, beispielsweise Converter-Abbildungstabellen, Collationsregeln, Transliterationsregelen, Regeln für Umbruch-Iteratoren und Diktionäre und sonstige Locale-Daten. Die ICU Data Library für Natural wird als Paket (Package) zur Verfügung gestellt, das die gewünschten Datenelemente enthält. Die Verwendung von Paketen anstelle von einzelnen Datenelementdateien steigert die Performance, weil während der Initialisierung nur ein Dateizugriff zum Laden des Pakets erfolgt. Es ist jedoch nicht so flexibel, denn es erfordert einen Rebuild des Pakets, wenn Datenelemente hinzugefügt werden müssen.

Die ICU Data Library kann kundenspezifisch angepasst werden, um existierende oder neue Converter-Abbildungstabellen oder um andere Datenelemente, z.B. Collationsregeln, Regeln für Umbruch-Iteratoren und sonstige Locale-Daten hinzuzufügen.

Das Tool für die kundenspezifische Anpassung der ICU Data Library kann aus dem Download Components-Bereich in Empower (***https://empower.softwareag.com/***) heruntergeladen werden. Benutzen Sie die mitgelieferte *icudtvr.zip*-Datei (*vr* = **Version**), um die Data Libraries für die ICU-Version anzupassen, die in Ihrer Natural-Umgebung erforderlich ist: Für ICS Version 2.2 ist *icudt58.zip* erforderlich. Die in diesem Abschnitt beschriebenen Dateien sind in der *icudtvr.zip*-Datei enthalten.

Um ein neues ICU Data Library Package zu erstellen, sind mehrere Schritte erforderlich. Einige davon werden auf einem PC durchgeführt, andere auf der zugehörigen Großrechner-Plattform.

- Neue Datenelemente beschaffen
- Konvertierungstabellen und Locales kompilieren
- Neues Data Library-Paket erstellen
- Assembler Source auf die Großrechner-Plattform übertragen

- Neue Data Library auf der Ziel-Großrechner-Plattform benutzen

Neue Datenelemente beschaffen

Für neue Datenelemente gibt es verschiedene Quellen:

- Der ICU Data Library Customizer auf ***http://apps.icu-project.org/datacustom/***.
- Das ICU Converter Data Archive auf ***http://source.icu-project.org/repos/icu/data/trunk/charset/data/ucm/***.
- Benutzerdefinierte Converter-Abbildungstabellen.

Der ICU Data Library Customizer ist ein web-basiertes Tool, das von IBM zur Verfügung gestellt wird. Für jede unterstützte Version ist ein ICU Data Library Customizer verfügbar. Die ICU-Version kann mit Hilfe der `SYSCP`-Utility festgestellt werden (siehe Funktion **ICU Information**).

Der Data Library Customizer zeigt die Datenelemente in einer Baumstrukturansicht an. Zunächst sind alle Datenelemente als ausgewählt markiert. Man kann alle Elemente zusammen abwählen, indem man die **Advanced**-Option erweitert und die Schaltfläche **Deselect All** wählt.

Die Menge der angezeigten Elemente kann reduziert werden, wenn man bei den **Advanced**-Optionen die Schaltfläche **Filter Items** wählt. Um zum Beispiel die Baumstrukturansicht so einzuschränken, dass nur Elemente für Japanisch angezeigt werden, muss man die Zeichenkette `japanese` in das Texteingabefeld eingeben und die Schaltfläche **Filter Items** wählen. Danach können mehrere Elemente ausgewählt oder abgewählt werden. Alle ausgewählten Elemente werden später zur ausgelieferten ICU Data Library hinzugefügt und stehen dann für Natural zur Verfügung.

Die zweite Möglichkeit besteht darin, für den gewünschten Converter eine (Source) Mapping-Datendatei *.ucm* zu übernehmen oder zu erstellen. Das ICU-Team unterhält ein umfangreiches Archiv mit Converter-Daten. Dieses Archiv ist versionsunabhängig. Falls die gewünschte Konvertierungstabelle nicht im Archiv vorhanden ist, kann man sich eine neu erstellen. Das Layout von Converter Mapping-Tabellen (*.ucm*-Dateien) ist im Kapitel *Conversion Data* im *ICU User Guide* dokumentiert, siehe ***http://userguide.icu-project.org/conversion/data***. Es wird empfohlen, aus dem Archiv eine ähnliche Mapping-Tabelle zu kopieren und sie an die neuen Erfordernisse anzupassen.

Anmerkung: Es ist untersagt, die Zeichenabbildung eines existierenden Converters zu ändern. Tatsächlich handelt es sich dabei um das Erstellen eines neuen Converters, was die Vergabe eines neuen Namens erfordert, um Verwechslungen zu vermeiden.

Ausführliche Informationen zur kundenspezifischen Anpassung der ICU Data Library finden Sie in der Datei *readme.txt*, die Teil der heruntergeladenen zip-Datei ist.

Konvertierungstabellen und Locales kompilieren

Converter-Quelldateien werden mit dem Tool *makeconv.exe* in binäre Converterdateien (*.cnv*-Dateien) kompiliert. Es ist möglich, mehr als einen Converter anzugeben:

Beispiel:

Kommando	Beschreibung
`makeconv ↩ ibm-1142_P100-1997.ucm`	Kompiliert die dänische Zeichenabbildungstabelle der Codepage IBM-1142 in das binäre Format. In einem anschließenden Schritt kann die Ausgabedatei *ibm-1142_P100-1997.cnv* zu dem neuen Data Library-Paket hinzugefügt werden.

Converters, die vom ICU übernommen werden, sind bereits in der Tabelle der Aliasnamen registriert. Falls die Converter-Source-Datei eine benutzerdefinierte Datei ist, existiert noch kein zugehöriger Name in der Tabelle der Aliasnamen. In diesem Fall müssen Sie die neue Codepage in der Tabelle der Aliasnamen registrieren. Öffnen Sie die Textdatei *convrtrs.txt* und fügen Sie einen entsprechenden Eintrag am Ende der Datei im Abschnitt "User defined code pages" (benutzerdefinierte Codepages) hinzu. Der Name der Codepage muss angegeben werden, die Angabe des IANA-Namens ist optional. Die Zeichenkette `{ IANA* }` deklariert *`iana-name`* als IANA-Namen. Für jede benutzerdefinierte Codepage muss ein Eintrag in der Tabelle der Aliasnamen vorgenommen werden.

Der Eintrag hat folgendes Format:

```
name-of-code-page  iana-name { IANA* }
```

Wenn die Codepage "my_cp-100" mit dem IANA-Namen "MYCP" hinzugefügt werden soll, ist in *convrts.txt* die folgende Zeile erforderlich:

```
my_cp-100            MYCP { IANA*}
```

Weitere Informationen finden Sie im Kopfdatenbereich der Datei *convrtrs.txt* oder im *ICU User Guide*.

Die geänderte Tabelle muss mit *gencnval.exe* in eine binäre Datei (*cnvalias.icu*) kompiliert werden, um sie dann mit dem neuen Data Library-Paket zu verlinken.

Neues Data Library-Paket erstellen

Ein neues Data Library-Paket wird mit dem Tool *makpkg.bat* erstellt. Es benutzt das gelieferte Paket *icudtvr.dat* (*vr* = **Version**) und sortiert neue, benutzerdefinierte Elemente ein. Ein benutzerdefiniertes Element kann ein zusätzliches Paket sein, das neue Datenelemente enthält, ein einzelnes Datenelement, z.B. ein neuer Converter (*.cnv*-Datei), oder eine Textdatei, die eine Liste neuer Elemente enthält.

Beispiele:

Kommando	Beschreibung
`makepkg icudtvrl.dat`	Fügt die Datenelemente, die in *icudtvrl.dat* enthalten sind, zum Basispaket *icudtvrb.dat* hinzu.
`makepkg ibm-1142_P100-1997.cnv`	Fügt die dänsiche Codepage IBM-1142 zum Basis-Data-Library-Paket *icudtvrb.dat* hinzu.
`makepkg newitems.txt`	Fügt alle Datenelemente (Converters), die in der Textdatei *newitems.txt* aufgeführt sind, zum Basis-Data-Library-Paket *icudtvrb.dat* hinzu.

makepkg.bat erzeugt zwei Dateien: eine Big-Endian EBCDIC-basierte binäre Datei und eine HL Assembler Source. Die Assembler Source enthält das binäre Abbild der ersten Datei, gepackt in `DC X'...'`-Statements. Der Name der binären Datei *icudtvre.dat* und der Name der Assembler Source ist *icudtvre_dat.s*. Die Datei *icudtvre* ist eine Kopie von *icudtvre_dat.s*. Die Dateien dürfen niemals umbenannt werden, weil der Paketname "icudt*vr*e" als Bestandteil interner Referenzen bei Datenelementen verwendet wird. Sie wird von der ICU Runtime für den Zugriff auf Datenelemente, z.B. Converters, und zum Validieren der Datendatei benutzt. "icudt" identifiziert die Datei als Big-Endian EBCDIC-kodiert.

Wenn mehr als ein Element hinzugefügt werden soll oder wenn die Tabelle der Aliasnamen geändert worden ist, müssen die Elemente als Liste in der Datei *newitems.txt* deklariert werden.

Beispiele:

- **Hinzufügen der Codepages ibm-939_P120-1999 und ibm-942_P12A-1999**
  Einträge in *newitems.txt*:

  ibm-939_P120-1999.cnv
  ibm-942_P12A-1999.cnv

- **Hinzufügen der benutzerdefinierten Codepage my_cp-100**
  Einträge in *newitems.txt*:

  cnvalias.icu
  my_cp-100.cnv

Weitere Informationen siehe *ICU User Guide*.

Assembler Source auf die Großrechner-Plattform übertragen

Das Ergebnis der vorangegangen Schritts ist ein Assembler-Modul. Das Assembler-Modul mit dem neuen Data Library-Paket *icudtvre* muss auf die Zielplattform übertragen werden. Das File Transfer Protocol (FTP) ist auf jedem PC vorhanden und kann für diesen Zweck benutzt werden. Wenden Sie sich an den Systemadministrator wegen der Informationen (z.B. Host-Name, Port-Nummer, Benutzername und Passwort), die Sie benötigen, um Zugang zur Zielmaschine über FTP zu erhalten. Da es sich bei *icudtvre* um eine Textdatei handelt, muss der Transfermodus auf ASCII gesetzt werden, um die korrekte Umsetzung der Datei auf der Zielplattform sicherzustellen. Der Name der Datei auf der Zielplattform kann frei gewählt werden. Es wird jedoch empfohlen, den Namen *icudtvre* zu verwenden. Falls eine Umbenennung der Datei *icudtvre* gewünscht wird, muss das Tool *renamepkg.bat* benutzt werden.

Neue Data Library auf der Ziel-Großrechner-Plattform benutzen

Das Assembler-Source-Modul muss auf der Zielplattform assembliert und gelinkt werden. Es kann entweder mit dem Nukleus verlinkt oder dynamisch geladen werden mit `RCA=name` und `CFICU=(DATFILE=name)`.

Natural-Profilparameter und Parameter-Makros

In diesem Abschnitt sind die Natural-Profilparameter und Parameter-Makros aufgeführt, die im Zusammenhang mit der Unicode- und Codepage-Unterstützung benutzt werden.

Falls nicht anderes angegeben ist, werden die in diesem Abschnitt erwähnten Profilparameter und Makros in der *Parameter-Referenz*-Dokumentation ausführlich beschrieben.

Parameter oder Makro	Beschreibung
`CFICU` oder Makro `NTCFICU`	Aktiviert die Unicode-Unterstützung für verschiedene Unicode-Einstellungen. Siehe auch *`CFICU`*-Parameter und ***`CFICU` and `CP`: Session-Modi***
`CMPO` oder Schlüsselwort-Subparameter `CPAGE` des Makros `NTCMPO`	Generiert codepage-sensitive Natural-Programme. Siehe auch ***`CPAGE`-Compiler-Option***.
`CP`	Definiert die Standard-Codepage für Natural. Diese Codepage wird für die Laufzeit- und die Entwicklungsumgebung benutzt, falls die Angabe nicht mit einer Codepage überschrieben wird, die für ein einzelnes Objekt (z.B. ein Natural-Quellcode-Objekt) definiert ist. Es können nur Codepages benutzt werden, die zur Plattform passen. Das bedeutet, dass für eine Großrechner-Plattform keine ASCII-Codepage definiert werden kann. Falls eine falsche Codepage benutzt wird, erscheint bei der Initialisierung eine Fehlermeldung.

Parameter oder Makro	Beschreibung
	Siehe auch *CFICU and CP: Session-Modi.*
CPCVERR	Gibt an, ob eine Fehlermeldung angezeigt wird, wenn ein Umsetzungsfehler bei folgenden Umsetzungen auftritt: von Unicode nach Codepage oder von Codepage nach Unicode oder von einer Codepage in eine andere Codepage. Dieser Parameter wird nicht bei der Umsetzung von Natural-Sourcen berücksichtigt, wenn diese in den Editierbereich geladen oder katalogisiert werden. Es wird nicht berücksichtigt, ob ein Unicode-Feld vor einer Eingabe/Ausgabe über eine Terminalemulation in die Codepage umgesetzt wird. In diesem Fall wird das durch ICU definierte Ersatzzeichen durch das in NATCONFG definierte Platzhalterzeichen ersetzt.
CPOBJIN	Dient zur Angabe der Codepage, in der die Batch-Eingabedatei codiert ist. Diese Datei ist im Dataset CMOBJIN definiert.
CPPRINT	Dient zur Angabe der Codepage, in der die Batch-Ausgabedatei codiert ist. Diese Datei ist im Dataset CMPRINT definiert.
CPSYNIN	Dient zur Angabe der Codepage, in der die Batch-Eingabedatei für Kommandos codiert ist. Diese Datei ist im Dataset CMSYNIN definiert.
NTCPAGE-Makro	Dieses Makro definiert im Modul NATCONFG eine Codepage und alle zugehörigen Informationen, z.B. Platzhalterzeichen, Locale ID und Collation Tables. Siehe auch *Makro NTCPAGE*. NTCPAGE und NATCONFG werden ausführlich in der *Operations*-Dokumentation behandelt.
OPRB oder NTOPRB-Makro	Dient zum Setzen der Optionen ACODE und/oder WCODE zur Definition der Benutzer-Zeichencodierung, wenn die benutzte Adabas-Datenbank für Universal Encoding Support (UES) freigegeben ist.
PRINT oder CP-Schlüsselwort-Parameter im Makro NTPRINT	Definiert die Codepage für einen Report.
SRETAIN	Dient zur Angabe, dass alle existierenden Quellcode-Objekte in ihrem Original-Zeichencodierungsformat gespeichert werden müssen. Siehe auch *Kundenspezifische Anpassung Ihrer Umgebung*.

Siehe auch:

- *Natural in Batch Mode* in der *Operations*-Dokumentation.
- Informationen zu gültigen Codepages siehe *http://www.iana.org/assignments/character-sets*.

  - Profilparameter CFICU
  - CFICU und CP: Session-Modi
  - Compiler-Option CPAGE

- Parameter-Makro NTCPAGE
- Natural Development Server (NDV)

Profilparameter CFICU

Eine ausführliche Beschreibung des Profilparameters `CFICU` und seiner Schlüsselwort-Subparameter befindet sich in der *Parameter-Referenz*-Dokumentation. Einige der Schlüsselwort-Subparameter haben Einfluss auf die Performance.

Wenn Collation Services zum Vergleichen von Unicode-Zeichenketten benutzt werden, werden beide Zeichenketten untersucht, ob sie normalisiert sind oder nicht. Die Prüfung selbst verbraucht viel CPU-Zeit. Wenn Sie sicher sind, dass die Zeichenketten bereits normalisiert sind, können Sie die Prüfung ausschalten (`COLNORM=OFF`).

In Unicode ist es möglich, dasselbe Zeichen als einen Codepoint oder als eine Kombination von zwei oder mehr Codepoints darzustellen. Beispielsweise kann das deutsche Zeichen "ä" durch "U+00E4" oder durch die Kombination der Codepoints "U+0061" und "U+0308" dargestellt werden. Bei der Umwandlung von Unicode nach beispielsweise IBM01140 werden Zeichen als einzelne Codepoints behandelt, d.h. es wird ein "a" gefolgt von einem Ersatzzeichen erzeugt, weil der Codepoint "U+0308" in der Ziel-Codepage nicht repräsentiert ist. Die Einstellung `CNVNORM=ON` bewirkt, dass vor der eigentlichen Umwandlung eine Normalisierung durchgeführt wird. Die Umwandlung verbraucht zusätzliche CPU-Zeit und Zwischenspeicherplatz. Wenn Sie sicher sind, dass bei `MOVE`-Statements (außer `MOVE NORMALIZED`) keine zusammengesetzten Zeichen beteiligt sind, sollten Sie `CNVNORM` auf `OFF` setzen, um die Performance zu verbessern. Beachten Sie, dass alle möglichen Kombinationen durch einen einzeln kodierten Unicode-Codepoint repräsentiert werden.

Die Umwandlung von Unicode in Codepage und umgekehrt erfordert keine hohe Leistung. Dies liegt daran, dass die ICU-Implementierung in C++ geschrieben ist, das nahezu alle Unicode-, Codepage- und Sprachenaspekte in der Welt abdeckt. Manche Codepages können jedoch über Umsetzungstabellen (Translation Tables) in Unicode (und umgekehrt) abgebildet werden, um die Umwandlung zu beschleunigen. Mit dem Schlüsselwort-Subparameter `CPOPT` des Profilparameters `CFICU` werden zur Beschleunigung Accelerator Tables aktiviert. Ist der Subparameter auf `ON` gesetzt, erstellt Natural unter Benutzung von ICU-Konvertierungsfunktionen bei der Session-Initialisierung zwei Accelerator Tables. Die erste Tabelle (mit einer Größe von 512 Bytes) wird für die Umwandlung von Codepage in Unicode und die andere Tabelle (mit einer Größe von 65535 Bytes) für die Umwandlung von Unicode in Codepage benutzt. Während der Natural-Session werden dann alle Umwandlungen über Accelerator Tables statt mit ICU-Aufrufen ausgeführt. Accelerator Tables werden nur für die Standard-Codepage (`*CODEPAGE`) verfügbar gemacht. Temporäre Codepages (z.B. in `MOVE ENCODED`-Statements) benutzen keine Accelerator Tables, wenn das Modul `NATCPTAB` nicht verlinkt ist. Wenn es verlinkt ist, werden bis zu 30 auf der ICU-Datenbank basierende Accelerator Tables benutzt, um die Performance zu beschleunigen.

CFICU und CP: Session-Modi

Die Profilparameter CFICU und CP können Sie benutzen, um Natural für bestimmte Zwecke anzupassen:

Einstellungen	Beschreibung
CFICU=OFF, CP=OFF	Kompatibilitätsmodus, um existierende Anwendungen ohne Unicode- und ohne Codepage-Unterstützung auszuführen. Für die Umsetzung der Ein- und Ausgaben werden herkömmliche Umsetzungstabellen verwendet. Im Vergleich zu früheren Versionen tritt kein signifikanter Ressourcenverbrauch (CPU-Zeit und Puffernutzung) auf. In diesem Modus braucht das ICS-Modul SAGICU (oder ein alternatives ICS-Modul) nicht mit dem Natural-Nukleus verlinkt zu sein.
CFICU=ON, CP=OFF	Für neue Anwendungen, bei denen Unicode- und Codepage-Umwandlung (MOVE ENCODED), aber keine Standard-Codepage-Unterstützung verwendet wird. Deshalb ist die Systemvariable *CODEPAGE leer. Es ist möglich, Variablen im Format U zu benutzen, aber es ist nicht möglich, z.B. MOVE A TO U zu benutzen, weil dafür die Standard-Codepage-Information erforderlich ist. Der Fehler NAT3411 wird ausgegeben, um anzuzeigen, dass keine Standard-Codepage verfügbar ist.
CFICU=ON, CP=*value* *	Für neue Anwendungen, bei denen vollständige Unicode- sowie Codepage-Unterstützung verwendet wird.
CFICU=OFF, CP=*value**	Diese Kombination ist nicht sinnvoll, weil zur Codepage-Unterstüzung ICU Services für die Umwandlung benötigt werden. Deshalb wird in diesem Fall die Einstellung CFICU=ON erzwungen und bei der Session-Initialisierung eine Meldung ausgegeben.

* dabei ist *value* ein beliebiger Wert ungleich OFF.

Compiler-Option CPAGE

Die Compiler-Option CPAGE erstellt Objekte, die mit einer anderen Codepage als die zur Erstellungszeit verwendete Codepage ausgeführt werden können. Das bedeutet, dass alle alphanumerischen Konstanten des Objekts, die in der zur Erstellungszeit verwendete Codepage kodiert sind, in die Codepage umgesetzt werden müssen, die zur Ausführungszeit aktiv ist. Um es dem Natural Object Loader zu ermöglichen, alphanumerische Konstanten zu finden und umzusetzen, wird vom Compiler eine zusätzliche Tabelle erstellt. Dadurch nimmt die Größe des generierten Objekts in Abhängigkeit von der Anzahl der verwendeten alphanumerischen Konstanten zu. Die Umwandlung zur Laufzeit verbraucht zusätzliche CPU-Zeit. Wenn die Standard-Codepage (Wert der Systemvariablen *CODEPAGE) dieselbe ist wie die zur Erstellungszeit verwendete Codepage oder wenn die Session keine Standard-Codepage hat (CP=OFF), wird keine Umwandlung durchgeführt. Unabhängig von der Einstellung des Profilparameters CPCVERR werden Umsetzungsfehler ignoriert. Wenn die Compiler-Option CPAGE auf OFF gesetzt ist, wird zur Laufzeit keine Umwandlung durchgeführt und die alphanumerischen Konstanten werden so behandelt, wie sie sind.

Das folgende Beispiel-Programm ist mit der Codepage IBM01141 (German) katalogisiert und wird mit der Standard-Codepage IBM01140 (us) ausgeführt. Die Zeichen "Ä", "Ö" und "Ü" sind in beiden Codepages definiert, jedoch an verschiedenen Codepoints.

Beispiel 1 - `CPAGE=OFF`:

```
OPTIONS CPAGE=OFF
WRITE *CODEPAGE  'ÄÖÜ'
END
```

Ausgabe mit Codepage IBM01140 (us):

```
Page      1

IBM01140                                                        ¢\!
```

Beispiel 2 - `CPAGE=ON`:

```
OPTIONS CPAGE=ON
WRITE *CODEPAGE  'ÄÖÜ'
END
```

Ausgabe mit Codepage IBM01140 (us):

```
Page      1

IBM01140                                                        ÄÖÜ
```

Parameter-Makro NTCPAGE

Am gebräuchlichsten für Codepage-Namen ist der IANA-Name. Deshalb enthält die Systemvariable `*CODEPAGE` den IANA-Namen der Standard-Codepage. Bei z/OS wird eine Codepage durch ihre Coded Character Set ID (CCSID) qualifiziert. Adabas verwendet zurzeit die Entire Conversion Service Definition (ADAECS).

Das Parameter-Makro `NTCPAGE` kann benutzt werden, um diese verschiedenen Namen zu dem eindeutigen IANA-Namen zuzuweisen. `NTCPAGE` ist Teil des Natural-Konfigurationsmoduls (`NATCONFG`). Es spielt keine Rolle, ob der IANA-Name, die CCSID/CCSN oder der Aliasname beim Profilparameter `CP` angeben wird. Der Aliasname kann ein benutzerdefinierter Name sein, der verwendet wird, um der Codepage einen aussagefähigeren Namen zuzuweisen. In jedem Fall enthält die Systemvariable `*CODEPAGE` den IANA-Namen der gewählten Codepage.

Zusätzlich kann für eine Codepage ein Platzhalterzeichen definiert werden. Es überschreibt das Standard-Ersatzzeichen der betreffenden Codepage, das normalerweise ein nicht anzeigbarers Zeichen ist (z.B. `H'3F'` in einer EBCDIC-Codepage). Das Platzhalterzeichen kann benutzt werden, um zu vermeiden, dass nicht anzeigbarer Zeichen an Terminals gesendet werden.

Beispiel:

```
NTCPAGE IANA=IBM01140,CCSID=1140,ECS=1140,ALIAS='US',PHC=003F
```

Die Werte `IBM01140`, `1140` oder `US` können mit dem Profilparameter `CP` eingegeben werden, um die Codepage zu aktivieren. Die Systemvariable `*CODEPAGE` enthält den Namen IBM01140. Das Ersatzzeichen der Codepage wird durch "U+003F" ersetzt, was ein Fragezeichen (?) ist

Die Anzahl der verfügbaren Codepages ist abhängig von der verwendeten ICU Data Library.

Alle im zurzeit verwendeten Datenpaket definierten Codepages können von Natural benutzt werden. Ein Eintrag in `NTCPAGE` ist nur nötig, wenn ein alternativer Aliasname oder ein alternatives Platzhalterzeichen gewünscht wird.

Natural Development Server (NDV)

Beim Natural Development Server (NDV) steht der folgende Konfigurationsparameter zur Verfügung:

Einstellung	Beschreibung
`TERMINAL_EMULATION=WEBIO`	Gibt an, welcher Natural Web I/O Interface Client (der Unicode unterstützt) für die Ein- und Ausgabe benutzt wird.

Information zur Zeichencodierung

Die Codepage-Information des Objekts ist Teil des Objektverzeichnisses, das mit dem Systemkommando `LIST` angezeigt werden kann. Weitere Informationen siehe *Directory-Informationen anzeigen* in der *Systemkommandos*-Dokumentation.

Die Zeichenkodierung von Codepage-Daten kann auf verschiedenen Ebenen angegeben werden.

Ebene 1 - Standard-Codepage

Die Standard-Codepage kann mit dem Natural-Profilparameter `CP` (Name der Standard-Codepage) angegeben werden.

Ebene 2 - Codepage für ein einzelnes Objekt

Eine Codepage kann angegeben werden für: Natural-Quellcode-Objekte, Batch-Eingabedateien (`CPOBJIN` - Codepage der Batch-Eingabedatei, `CPSYNIN` - Codepage der Batch-Eingabedatei für Kommandos) und Ausgabedateien (`CPPRINT` - Codepage der Batch-Ausgabedatei).

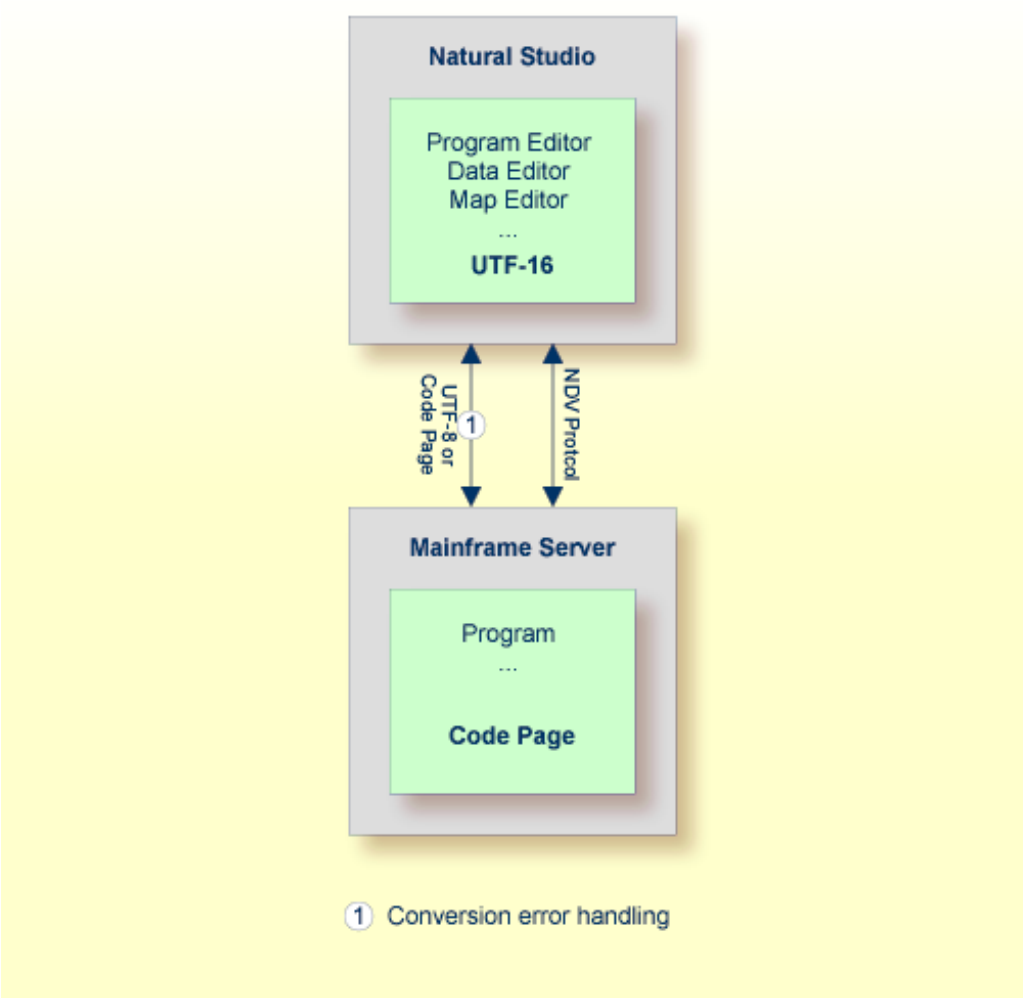
Die Definition einer Codepage auf der Objektebene ist maßgeblich für das betreffende Objekt und hat dort Vorrang vor der Standard-Codepage.

6 Entwicklungsumgebung

- Entwicklungsumgebung für Anwendungen 34
- Kundenspezifische Anpassung Ihrer Umgebung 35
- Editoren in der SPoD-Umgebung 35
- Codepage-Unterstützung bei Editoren, Systemkommandos und Utilities 36
- Codepage-Unterstützung bei Natural-Quellcode-Objekten 39

Entwicklungsumgebung für Anwendungen

Der Natural Single Point of Development (SPoD) ist die Entwicklungsumgebung für Unicode-Anwendungen.



In einer SPoD-Umgebung können die Natural-Objekte, die auf einem Natural Development Server (NDV) liegen, mittels Natural Studio geändert werden. Falls vom Server unterstützt, werden die Quellcode-Objekte im Format UTF-8 zwischen Client und Server ausgetauscht.

Auf NDV-Servern werden die Objekte mit der Standard-Codepage oder mit ihrer Original-Zeichencodierung gespeichert. Dies ist abhängig von der Einstellung des Profilparameters `SRETAIN`.

Kundenspezifische Anpassung Ihrer Umgebung

Wenn der Profilparameter `SRETAIN` auf `OFF` gesetzt ist, werden alle Quellcode-Objekte mit der Standard-Codepage gespeichert.

Bei dieser Einstellung ist Vorsicht geboten, weil sie zu inkorrekten Codepage-Informationen führen kann, wenn Sie Quellcode-Objekte haben, die mit einer früheren Natural-Version erstellt wurden. In diesem Fall ist die Zeichencodierungsinformation des Quellcode-Objekts nicht zugewiesen und das Quellcode-Objekt wird immer mit der Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) geöffnet. Dies wird oft funktionieren, selbst wenn die Standard-Codepage nicht die korrekte Zeichencodierung des Quellcode-Objekts ist. In diesem Fall werden manche sprachspezifischen Zeichen falsch angezeigt. Wenn ein solches Quellcode-Objekt mit der falschen Codepage geöffnet wird und wenn beim Speichern der Profilparameter `SRETAIN` auf `ON` gesetzt ist, dann wird für das Quellcode-Objekt keine Zeichensatzcodierung gespeichert; das Quellcode-Objekt kann später korrekt geöffnet werden, wenn Natural mit der korrekten Standard-Codepage gestartet wird. Wenn Sie jedoch einmal das Quellcode-Objekt mit `SRETAIN` auf `OFF` gespeichert haben, wird die Standard-Codepage als die Zeichencodierung des Quellcode-Objekts gespeichert; von da an wird das Quellcode-Objekt immer mit dieser Codepage geöffnet.

Deshalb sollten Sie diese Einstellung nur benutzen, wenn Sie sicher sind, dass alle Ihre Natural-Quellcode-Objekte in der Standard-Codepage kodiert sind.

Editoren in der SPoD-Umgebung

Editoren, die in Natural for Windows sind vollständig Unicode-fähig. Über SPoD können Sie auch für Großrechner-Quellcode-Objekte benutzt werden. Die in Natural für Großrechner vorhandenen Editoren sind nicht Unicode-fähig.

Anmerkung: Die in Natural für Großrechner vorhandenen Editoren bieten Codepage-Unterstützung. Siehe ***Codepage-Unterstüzung bei Editoren, Systemkommandos und Utilities***.

Wenn ein Quellcode-Objekt in einem Editor in Natural Studio (Natural for Windows) geöffnet wird, wird der Inhalt des Quellcode-Objekts von der entsprechenden Codepage nach Unicode umgewandelt, bevor es in den Editor geladen wird. Dadurch wird sichergestellt, dass alle Zeichen sogar dann korrekt angezeigt werden können, wenn das Quellcode-Objekt Zeichen enthält, die in der System-Codepage nicht enthalten sind. Falls die Umwandlung von der Codepage des Quellcode-Objekts nach Unicode fehlschlägt, wird ein Fehler angezeigt und der Editor wird nicht geöffnet. In diesem Fall muss der Benutzer die korrekte Zeichencodierung für das Quellcode-Objekt definieren. Die Quellcode-Objekt-Zeichencodierung (Encoding) kann im Dialogfenster **Properties** (Eigenschaften) geändert werden (siehe *Properties for the Nodes* in der *Using Natural Studio*-Dokumentation).

Mit dem Programm-Editor in Natural for Windows können Sie Textkonstanten in ihre hexadezimale Unicode-Darstellung umwandeln (siehe *Converting to Hexadecimal Format* im Kapitel *Program Editor* der Natural for Windows *Editors*-Dokumentation). Wenn Sie für eine Plattform entwickeln, auf der Quellcode-Objekte im Format UTF-8 nicht bevorzugt werden, können Sie so alle Zeichen für eine Unicode-Konstante eingeben, alle Zeichen der Konstanten auswählen, sie in hexadezimale Darstellung umwandeln und das Präfix "UH" für hexadezimale Unicode-Konstanten hinzufügen. Und wenn Sie den Mauszeiger eine Weile über ein Zeichen oder einen ausgewählten Zeichenbereich einer Textkonstante halten, wird in einem Tool-Tip-Fenster die entsprechende hexadezimale Unicode-Darstellung angezeigt.

Codepage-Unterstützung bei Editoren, Systemkommandos und Utilities

- Natural-Editoren
- Natural-Systemkommandos und Utilities

Natural-Editoren

Der Programm-Editor, der Masken-Editor (Map Editor) und der Datenbereich-Editor (Data Area Editor) sind nicht Unicode-fähig. Stattdessen werden Quellcode-Objekte mit Codepage-Informationen gespeichert. Je nach Einstellung des Profilparameters `SRETAIN` können Natural-Quellcode-Objekte mit Codepage-Informationen automatisch von der aktuellen Codepage des Quellcode-Objekts in die Standard-Codepage der aktuellen Natural-Session (Wert der Systemvariablen `*CODEPAGE`) umgewandelt werden, wenn der Quellcode in den Editor geladen wird. Falls es Zeichen gibt, die nicht umgewandelt werden können, wird in einem Fenster ein Codepoint-Konvertierungsfehler angezeigt und die Angabe von Ersatzwerten für die nicht umwandelbaren Codepoints angefordert. Die Anzeige dieser Meldung ist unabhängig von der aktuellen Einstellung des Profilparameters `CPCVERR`. In diesem Fall kann sich der Benutzer entscheiden, den Editor mit oder ohne Umwandlung des Quellcodes in die Standard-Codepage zu öffnen. Beim Speichern (`SAVE`) oder Katalogisieren (Kompilieren) und Speichern (`STOW`) eines konvertierten Quellcodes wird die neue Codepage-Information gespeichert. Quellcodes ohne Codepage-Information (z.B. Quellcodes, die mit einer früheren Natural-Version gespeichert worden sind), werden ohne Umwandlung in die Editoren geladen. Entsprechend der Einstellung des Profilparameters `SRETAIN` wird die aktuelle Codepage-Information des Quellcodes beibehalten.

Das Einfügen von Quellcodes mittels des Kommandos `.I` oder der Split-Screen-Funktion (geteilter Bildschirm) bewirkt ebenfalls (falls nötig) eine Umwandlung von Quellcodes entsprechend der Einstellung des Profilparameters `SRETAIN`. Falls es Zeichen gibt, die nicht umgewandelt werden können, wird stattdessen das definierte Ersatzzeichen eingefügt.

Die Prüfung und Umwandlung des Quellcodes wird durchgeführt, wenn der Editor gestartet wird, nicht wenn das Programm in den Source-Bereich des Editors geladen wird. Falls ein Programm mit dem Kommando `RUN program-name` ausgeführt wird, erfolgt keine Umwandlung. Das verursacht ein unterschiedliches Verhalten, je nachdem ob das Kommando `RUN program-name`

im NEXT-Bildschirm oder in einem Editor-Bildschirm eingegeben wird. Wird RUN *program-name* im NEXT-Bildschirm eingegeben, dann erfolgt keine Umwandlung. Wird das Kommando in einem Editor-Bildschirm eingegeben, wird der Editor unmittelbar nach der Ausführung des Programms gestartet und es wird eine Umwandlung durchgeführt.

Die folgende Tabelle zeigt, welche Codepage einer existierenden Natural Source zugewiesen wird, die gespeichert (SAVE) oder gespeichert und katalogisiert (STOW) wird, in Abhängigkeit von den Werten der Profilparameter SRETAIN und CP.

Original-Source-Codepage-Informationen	**Einstellung von** SRETAIN	**Source-Codepage-Informationen nach** SAVE **oder** STOW**, wenn** CP **auf einen Wert ungleich** OFF **gesetzt ist**	**Source-Codepage-Inform nach** SAVE **oder** STOW**, auf** OFF **gesetzt ist**
Source ohne Codepage-Informationen	SRETAIN=ON SRETAIN=(ON,EXCEPTNEW)	Keine Codepage-Informationen	Keine Codepage-Informatio
Source ohne Codepage-Informationen	SRETAIN=OFF	Codepage ergibt sich aus der Auswertung von CP	Keine Codepage-Informatio
Source ist kodiert in *code page 1*	SRETAIN=ON SRETAIN=(ON,EXCEPTNEW)	Original-Codepage (*code page 1*)	Original-Codepage (*code page 1*)
Source ist kodiert in *code page 1*	SRETAIN=OFF	Codepage ergibt sich aus der Auswertung von CP	Original-Codepage (*code page 1*)

Die folgende Tabelle zeigt, welche Codepage einer neuen Natural Source zugewiesen wird, die gespeichert (SAVE) oder gespeichert und katalogisiert (STOW) wird, in Abhängigkeit von den Profilparametern SRETAIN und CP.

Einstellung von SRETAIN	**Source-Codepage-Informationen nach** SAVE **oder** STOW**, wenn** CP **auf einen Wert ungleich** OFF **gesetzt ist**	**Source-Codepage-Informationen nach** SAVE **oder** STOW**, wenn** CP **auf** OFF **gesetzt ist**
SRETAIN=ON	Codepage ergibt sich aus der Auswertung von CP	Keine Codepage-Informationen
SRETAIN=OFF	Codepage ergibt sich aus der Auswertung von CP	Keine Codepage-Informationen

Einstellung von SRETAIN	Source-Codepage-Informationen nach SAVE oder STOW, wenn CP auf einen Wert ungleich OFF gesetzt ist	Source-Codepage-Informationen nach SAVE oder STOW, wenn CP auf OFF gesetzt ist
SRETAIN=(ON,EXCEPTNEW)	Keine Codepage-Informationen	Keine Codepage-Informationen

Natural-Systemkommandos und Utilities

Systemkommando LIST

Mit dem Systemkommando LIST wird Quellcode standardmäßig so angezeigt, wie er in der Systemdatei gespeichert ist, ohne Konvertierung.

Bei Auswahl der Option CONVERTED des LIST-Kommandos wird der Quellcode in die Standard-Codepage (Wert der Systemvariablen *CODEPAGE) umgesetzt, falls die Codepage-Informationen des Quellcodes verfügbar sind. Alle nicht konvertierbaren Zeichen werden dann durch das definierte Ersatzzeichen ersetzt.

Systemkommando LIST DIR

Das Systemkommando LIST DIR zeigt die Informationen der verwendeten Codepage eines Natural-Quellcodes in einem Directory-Fenster.

Systemkommando SCAN

Ähnlich wie die Editoren wandelt das Systemkommando SCAN die Quellcodes um, bevor die aktuelle Suche durch das SCAN-Kommando ausgeführt wird.

Object Handler (SYSOBJH Utility)

Der Object Handler entlädt oder lädt Quellcodes mit unterschiedlichen Codepage-Informationen und bewahrt die Original-Codepage-Informationen.

Nach Auswahl der Transfer-Format-Option UTF-8 werden Quellcodes beim Entladen aus einer beliebigen Codepage in das Format UTF-8 umgewandelt und Informationen über die Original-Codepage in der Arbeitsdatei gespeichert. Die entsprechende Ladefunktion wandelt den Quellcode zurück in die Original-Codepage (falls angegeben). Diese Option ermöglicht es auch, Codepage-Informationen für Quellcodes beizugeben, die mit früheren Natural-Versionen gespeichert (SAVE) oder gespeichert und katalogisiert (STOW) worden sind und deshalb keinerlei Codepage-Informationen enthalten.

Lade- und Entlade-Quellcodes in internem Format behalten die Codepage-Informationen (falls vorhanden).

Utility SYSCP

Die Utility `SYSCP` (siehe in der *Debugger und Dienstprogramme (Utilities)*-Dokumentation) kann benutzt werden, um Informationen zu Codepages zu erhalten und die Codepage-Zuweisung eines Quellcodes zu prüfen oder zu ändern.

Codepage-Unterstützung bei Natural-Quellcode-Objekten

Der Natural-Compiler, die Editoren und die Natural-Systemdatei unterstützen keinen Objekt-Quellcode, der in Unicode kodiert ist. Unicode-Konstanten, die in einem Objekt-Quellcode kodiert sind, werden in der Standard-Codepage gespeichert und das katalogisierte Objekt enthält die Unicode-Codepoints. Die einzige Möglichkeit, Unicode-Konstanten zu definieren, die kein Äquivalent in der Standard-Codepage haben, besteht darin, hexadezimale Definitionen (UH) zu benutzen.

Da Natural-Quellcodes vor dem Speichern nicht in Unicode oder UTF-8 umgesetzt werden, können Sie noch von früheren Natural-Versionen gelesen werden. Codepage-Informationen werden im Kopfdatenbereich des Quellcodes gespeichert. Die Codepage-Informationen im Kopfdatenbereich werden einfach ignoriert, wenn auf einen Quellcode mit einer Natural-Version zugegriffen wird, die nicht Unicode-fähig ist.

- Programme, Datenbereiche (Data Areas) und Masken (Maps)
- Datendefinitionsmodule (DDMs)
- Fehlermeldungen
- Hilfetexte

Programme, Datenbereiche (Data Areas) und Masken (Maps)

Der Quellcode dieser Objekte wird nicht im Unicode-Format, sondern in der Standard-Codepage der aktuellen Natural-Session gespeichert. Der Name der Codepage wird im Verzeichnis (Directory) des Quellcode-Objekts gespeichert. Deshalb bleibt die Größe des Quellcode-Objekts im Vergleich zu früheren Natural-Versionen unverändert. Es erfolgt jedoch eine Prüfung durch den Editor, ob die Codepage des Quellcodes gleich der Standard-Codepage der Natural-Session ist. Falls die Codepages verschieden sind, wird der Quellcode in die Standard-Codepage umgewandelt, wobei es zu Umsetzungsfehlern kommen kann. Wenn ein Zeichen im Quellcode nicht in der Standard-Codepage abgebildet ist, erscheint im Editor ein Fenster, das die manuelle Umwandlung der bei der Umsetzung fehlgeschlagenen Zeichen ermöglicht. Beispiel: Ein Programm, das mit der Codepage IBM01140 erstellt wurde, enthält folgende Zeile:

```
WRITE '100 €'
```

Wenn das Programm erneut bearbeitet wird, während Natural mit der Codepage IBM037 läuft, tritt ein Umwandlungsfehler auf, weil das Zeichen "€" in der Codepage IBM037 nicht abgebildet ist.

Zu beachten ist, dass die Umwandlung erfolgt, wenn der Editor gestartet wird, und nicht wenn der Quellcode geladen wird.

Datendefinitionsmodule (DDMs)

Datendefinitionsmodule (DDMs) werden nicht in Unicode-Format, sondern in der Standard-Codepage der aktuellen Nautural-Session gespeichert. Der Name der Codepage wird im Verzeichnis (Directory) des Datendefinitionsmoduls gespeichert. Zu beachten ist, dass es in der Systemdatei kein Datendefinitionsmodul gibt. Im Vergleich zu früheren Natural-Version ist das Datendefinitionsmodul etwas größer. Beim Lesen eines Datendefinitionsmoduls erfolgt eine Prüfung durch den Editor, ob die Codepage des Datendefinitionsmoduls gleich der Standard-Codepage der Natural-Session ist. Falls die Codepages verschieden sind, wird das Datendefinitionsmodul in die Standard-Codepage umgewandelt, wobei es zu Umsetzungsfehlern kommen kann. Wenn ein Zeichen im Datendefinitionsmodul nicht in der Standard-Codepage abgebildet ist, erscheint im Editor ein Fenster, das die manuelle Konvertierung der bei der Umsetzung fehlgeschlagenen Zeichen ermöglicht. Beispiel: Ein Datendefinitionsmodul, das mit der Codepage IBM01140 erstellt wurde, enthält folgende Zeile:

```
* 100 €
```

Wenn das Datendefinitionsmodul erneut bearbeitet wird, während Natural mit der Codepage IBM037 läuft, tritt ein Umwandlungsfehler auf, weil das Zeichen "€" in der Codepage IBM037 nicht abgebildet ist.

Fehlermeldungen

Natural-Fehlermeldungen werden nicht in Unicode-Format, sondern in der Standard-Codepage der aktuellen Natural-Session gespeichert. Der Name der Codepage wird in einem zusätzlichen Adabas-Feld in der Systemdatei gespeichert. Die Utility `SYSERR` bietet die Möglichkeit, zu prüfen, ob die Codepage der Fehlermeldung gleich der Standard-Codepage der Natural-Session ist. Falls die Codepages verschieden sind, wird die Fehlermeldung in die Standard-Codepage umgewandelt. Umsetzungsfehler werden ignoriert. Das bedeutet, es wird das Ersatzzeichen (oder, falls definiert, das Platzhalterzeichen) verwendet.

Hilfetexte

Hilfetexte werden immer in der Codepage IBM01140 (English) gepflegt. Es wird keine Codepage-Definition mit gespeichert. Falls die Standard-Codepage nicht IBM01140 ist, wird der Hilfe-Text in die Standard-Codepage umgesetzt. Umsetzungsfehler werden ignoriert. Es wird das Ersatzzeichen (oder, falls definiert, das Platzhalterzeichen) verwendet.

7 Unicode- und Codepage-Unterstützung in der Natural-Programmiersprache

- Natural-Datenformat U für Unicode-basierende Daten 44
- Natural-Statements 45
- Logische Bedingungen 50
- Systemvariablen 50
- Große und dynamische Variablen 51
- Session-Parameter 51
- Beispiel-Programme 54

Natural-Datenformat U für Unicode-basierende Daten

In der Natural-Programmiersprache können Sie Unicode-Zeichenketten mit Format U und U-Konstanten angeben.

- **Format U**
  Im Format U können Sie Daten definieren, die Unicode-Zeichenketten enthalten. Das Natural-Datenformat U ist intern UTF-16.

  Siehe auch *Format und Länge von Benutzervariablen* im *Leitfaden zur Programmierung*.

- **U-Konstante**
  Sie können Unicode-Konstanten mit dem Präfix "U" definieren. Zum Beispiel:

  ```
  U'Äpfel'
  ```

  Das Präfix "UH" kann benutzt werden, um Unicode-Konstanten in hexadezimalem Format zu definieren. Vier hexadezimale Ziffern stellen gemäß Unicode Standard eine UTF-16-Codeeinheit (Code Unit) dar. Daher muss die Gesamtlänge ein Vielfaches von Vier sein. Angenommen, Sie benötigen die hexadezimale Form von

  ```
  U'Äpfel'
  ```

  dann brauchen Sie die UTF-16-Codeeinheiten für "Ä", "p", "f", "e" und "l" (die sind "U+00C4", "U+0070", "U+0066", "U+0065" und "U+006C") und Sie müssen sie zu folgender hexadezimalen Zeichenkette zusammenfügen:

  ```
  UH'00C4007000660065006C'
  ```

  Siehe auch *Unicode-Konstanten* im *Leitfaden zur Programmierung*.

Das Datenformat U ist Endian-abhängig. Dies ist beim Übertragen (`MOVE`) zwischen den Formaten B und U zu berücksichtigen.

Format U im Vergleich zu A

Der Vorteil des Formats U (im Vergleich zum Format A) liegt darin, dass es beliebige Zusammenfügungen von Zeichen aus verschiedenen Sprachen enthalten kann und dass es nicht von der Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) abhängig ist. Darüber hinaus vereinfacht das Format U den Austausch von Daten zwischen unterschiedlichen Plattformen. Es sind keine Umwandlungen (z.B. von EBCDIC nach ASCII) nötig.

Andererseits wird für Daten im Format U mehr Speicherplatz als bei Daten im Format verbraucht. Bei Sprachen, die in Einzel-Byte-Zeichencodierung dargestellt werden können, führt das Format

U dazu, dass Zeichenketten doppelt so viel Platz belegen, wie früher erforderlich war. Bei ostasiatischen Sprachen dagegen ist der Speicherplatzverbrauch oft nicht größer.

Natural-Statements

Grundsätzlich gilt, dass das Format U in den meisten Statements benutzt werden kann, die das Format A erlauben. Wenn aber ein Natural-Objektname als Operand eines Statements gegeben ist (z.B. im `CALLNAT`-Statement), dann kann das Format U nicht benutzt werden, weil Natural-Objekte das Format A haben. Informationen zu bestimmten Statements können Sie der *Statements*-Dokumentation entnehmen.

Grundsätzlich können die Formate A und U zusammen in einem Statement benutzt werden, z.B.:

```
EXAMINE S FOR P WITH DELIMITER D REPLACE R
```

dabei ist `S` im Format U und `P`, `D` und `R` sind im Format A.

Im obigen Beispiel werden die Variablen `P`, `D` und `R` vor der tatsächlichen Ausführung des `EXAMINE`-Statements temporär in das Zielformat U umgewandelt. Die Umwandlung von Unicode nach Codepage oder umgekehrt erfordert den Aufruf einer ICU-Funktion. Diese Umwandlung beansprucht zusätzliche Rechenzeit und zusätzlichen Speicherplatz. Dieser Nachteil wird bei sehr großen Variablen noch größer. Um häufige Umwandlungen zu vermeiden, wird deshalb empfohlen, innerhalb eines Statements nur ein Format zu benutzen. Wenn alle Operanden im obigen Beispiel entweder im Format U oder im Format A angegeben werden, entfällt die Notwendigkeit einer Umwandlung. Wenn Sie sich jedoch dafür entscheiden, nur Operanden im Format U anzugeben, dann ist diese Variante (artbedingt) langsamer, weil dieser Operandentyp mehr Ressourcen verbraucht. Ein Zeichen wird dann mit 2 Bytes kodiert (anstelle von 1 Byte, das für das Format A benutzt wird).

Bei einer Umwandlung (insbesondere vom Format U ins Format A) besteht immer das Risiko, dass Zeichen nicht in der Ziel-Codepage dargestellt werden können. Angenommen Sie wollen das Unicode-Zeichen "U+05D0" (hebräischer Buchstabe Alef) in die Codepage IBM01140 (Englisch) umwandeln. Da dieses Zeichen in der Codepage IBM01140 nicht enthalten ist, wird entweder das Ersatzzeichen für diese Codepage benutzt oder der Platzhalter, der bei der Definition der Codepage im Modul `NATCONFG` angegeben worden ist. Wenn der Profilparameter `CPCVERR` auf `ON` gesetzt ist, wird in diesem Fall eine Fehlermeldung ausgegeben, die einen Umwandlungsfehler anzeigt. Die ursprüngliche Information geht auf jeden Fall verloren.

Bei der Benutzung von Unicode sind insbesondere die folgenden Statements betroffen:

- MOVE NORMALIZED
- MOVE ENCODED
- EXAMINE
- PARSE JSON

- PARSE XML
- REQUEST DOCUMENT
- DEFINE PRINTER
- CALLNAT (RPC)

MOVE NORMALIZED

Normalisierung in Unicode: Ein Vorgang zum Entfernen alternativer Darstellungen von äquivalenten Zeichenfolgen aus Textdaten, um die Daten in eine Form umzuwandeln, die binär auf Äquivalenz verglichen werden kann. Im Unicode Standard sind verschiedene Normalisierungsformen definiert. Die Normalisierungsform, die sich bei der kanonischen Zerlegung einer Unicode-Zeichenkette ergibt, und die im Anschluss daran (wo möglich) erfolgende Ersetzung aller zerlegten Zeichenfolgen durch Primärkompositen wird „Normalization Form Composed“ (NFC) genannt.

Natural nimmt an, dass alle Unicode-Daten im NFC-Format sind, um sicherzugehen, dass String-Operationen ohne teilweise Trunkierung eines Unicode-Zeichens durchgeführt werden können. Bei den Umwandlungsoperationen in Natural wird sichergestellt, dass die resultierenden Unicode-Zeichenkette in NFC ist. Wenn Unicode-Daten von Außerhalb von Natural empfangen werden und wenn nicht gewährleistet ist, dass die Daten NFC-Format haben, kann das Natural-Statement `MOVE NORMALIZED` angewendet werden.

Beispiel:

Zeichenfolge	NFC
ê (U+00EA)	ê (U+00EA)
e (U+0065) + ^ (U+0302)	ê (U+00EA)

Anmerkung: Wenn zwei oder mehr Zeichenketten im NFC-Format verkettet werden, kann es sein, dass das Resultat nicht im NFC-Format ist.

MOVE ENCODED

Eine implizite Umwandlung zwischen Unicode und der Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) wird durchgeführt, wenn Zeichenketten mittels `MOVE`-Statement von Format U nach Format A oder umgekehrt übertragen werden.

Weiterhin kann das `MOVE ENCODED`-Statement zur Umwandlung zwischen verschiedenen Codepages oder von einer beliebigen verfügbaren Codepage nach Unicode und umgekehrt benutzt werden. Dies kann hilfreich sein, wenn Daten von Außerhalb von Natural kommen und wenn diese Daten in einer Codepage, die ungleich zur Standard-Codepage ist, kodiert sind. Aber Sie können dieses Statement sogar zur Umwandlung zwischen der Standard-Codepage und Unicode benutzen, wenn Sie einen potenziellen Umwandlungsfehler mit der `GIVING`-Klausel erhalten wollen; wenn `CPCVERR` auf `ON` gesetzt ist, wird in diesem Fall das `MOVE`-Statement mit einem Laufzeitfehler stoppen.

Falls ein Zeichen nicht umgesetzt werden kann, hängt es von der Einstellung des `CPCVERR`-Parameters ab, ob ein Ersatzzeichen für dieses Zeichen benutzt wird oder ob die Umwandlung fehlschlägt.

Dieses Statement kann außerdem von U-Daten in das UTF-8-Format benutzt werden.

Anmerkung: Wenn Sie Daten in eine Codepage umwandeln, die ungleich der Standard-Codepage ist, wird empfohlen, diese Daten nicht bei Ein- und Ausgaben zu benutzen. Ein- und Ausgaben sind nur in der Standard-Codepage sinnvoll.

EXAMINE

Ein „Graphem“ ist das, was sich ein Benutzer normalerweise unter einem Zeichen vorstellt. In den meisten Fällen ist ein Unicode-Codepoint ein Graphem, aber ein Graphem kann auch aus mehreren Unicode-Codepoints bestehen.Zum Beispiel ist eine Folge von einem Basiszeichen und einem oder mehreren Kombinationszeichen ein Graphem.

Beispiel: "a" (U+0061) + "." (U+0323) + "^" (U+0302) definiert ein Graphem, das wie folgt angezeigt wird:

ậ

Anmerkung: Wenn eine Basis-/Kombinationszeichenfolge normalisiert wird, bedeutet dies nicht, dass die Folge immer durch ein zusammengesetzes (precomposed) Zeichen ersetzt wird, da nicht alle Zeichen im zusammengesetzten Format verfügbar sind.

Ein ergänzender Codepoint ist ein Unicode-Codepoint zwischen "U+10000" und "U+10FFFF". Ein ergänzender Codepoint wird in UTF-16 durch ein Surrogatpaar (Ersatzpaar) dargestellt, das aus zwei Codeeinheiten besteht, wobei der erste Wert des Paares eine „high-surrogate code unit“ und der zweite eine „low-surrogate code unit“ ist. Solche Zeichen sind im Allgemeinen selten, aber einige werden z.B. als Teil chinesischer und japanischer Personennamen verwendet, und daher ist die Unterstützung dieser Zeichen häufig für Regierungsanwendungen in ostasisatischen Ländern erforderlich.

Die Statements zur Behandlung von Zeichenketten, z.B. `EXAMINE` und die Option `SUBSTRING` arbeiten mit UTF-16-Codeeinheiten. Es liegt in der Verantwortung des Benutzer, dass der Code keine Grapheme oder Surrogatpaare trennt.

Die Klauseln `CHARPOSITION` und `CHARLENGTH` des `EXAMINE`-Statement (siehe *Syntax 3 - EXAMINE für Unicode-Grapheme*) können jedoch benutzt werden, um den Anfang und die Länge (in UTF-16-Codeeinheiten) von Graphemen abzufragen. Die Ergebniswerte können für `SUBSTRING`-Aufrufe verwendet werden. Mit diesen Klauseln ist es möglich, eine Zeichenkette Graphem für Graphem zu durchsuchen.

Beispiel:

```
DEFINE DATA LOCAL
1 #UNICODE-STRING    (U15)
1 #CODE-UNIT-INDEX   (N4)
1 #CODE-UNIT-LEN     (N4)
1 #GRAPHEME-NUMBER   (N4)
END-DEFINE

MOVE U'aक्षbācक्ष' TO #UNICODE-STRING

#GRAPHEME-NUMBER := 1

REPEAT
EXAMINE
        FULL VALUE OF #UNICODE-STRING
        FOR CHARPOSITION #GRAPHEME-NUMBER
        GIVING POSITION IN #CODE-UNIT-INDEX
        GIVING LENGTH IN #CODE-UNIT-LEN

        DISPLAY #UNICODE-STRING #GRAPHEME-NUMBER #CODE-UNIT-INDEX #CODE-UNIT-LEN

        #GRAPHEME-NUMBER := #GRAPHEME-NUMBER + 1
WHILE #CODE-UNIT-INDEX NE 0
END-REPEAT

END
```

Ausgabe des obigen Programms:

```
Page     1                                                    06-17-26  09:33:49

#UNICODE-STRING #GRAPHEME-NUMBER #CODE-UNIT-INDEX #CODE-UNIT-LEN
--------------- ---------------- ---------------- --------------
aक्षbācक्ष           1                    1                        1
aक्षbācक्ष           2                    2                        3
aक्षbācक्ष           3                    5                        1
aक्षbācक्ष           4                    6                        3
aक्षbācक्ष           5                    9                        1
aक्षbācक्ष           6                   10                        3
aक्षbācक्ष           7                    0                        0
```

PARSE JSON

Das zu parsende Dokument wird immer intern in UTF-8 gewandelt (wenn das Dokument noch nicht in UTF-8 kodiert ist).

Weitere Informationen siehe Beschreibung des Statements `PARSE JSON`.

Siehe auch *Statements für den Internet-Zugriff und Parsing* im *Leitfaden zur Programmierung*.

PARSE XML

Das zu parsende Dokument wird immer intern in UTF-16 gewandelt (wenn das Dokument noch nicht in UTF-16 kodiert ist).

Weitere Informationen siehe Beschreibung des Statements `PARSE XML`.

Siehe auch *Statements für den Internet-Zugriff und Parsing* im *Leitfaden zur Programmierung*.

REQUEST DOCUMENT

Bei der Übertragung von Daten mit dem `REQUEST DOCUMENT`-Statement erfolgt normalerweise keine Codepage-Umwandlung. Wenn Sie wollen, dass die ausgehenden und/oder eingehenden Daten in einer spezifischen Codepage zeichencodiert sind, können Sie die Klausel `DATA ALL` und/oder die Klausel `RETURN PAGE` des `REQUEST DOCUMENT`-Statements benutzen, um dies anzugeben.

Weitere Informationen siehe Beschreibung des Statements `REQUEST DOCUMENT`.

Siehe auch *Statements für den Internet-Zugriff und Parsing* im *Leitfaden zur Programmierung*.

DEFINE PRINTER

Das `DEFINE PRINTER`-Statement stellt eine `CODEPAGE`-Klausel zur Verfügung, mit der für die Umwandlung von Druckreportdaten in eine Codepage ungleich der Standard-Codepage (Wert der Systemvariablen `*CODEPAGE` gesorgt werden kann.

CALLNAT (RPC)

Der Austausch von Daten in U-Format via RPC wird unterstützt. Siehe Beschreibung des `CALLNAT`-Statements.

Wenn Daten im Format U von einer Plattform mit Big-Endian-Zeichencodierung an eine Plattform mit Little-Endian-Zeichencodierung oder umgekehrt gesendet werden, wird die Zeichencodierung so angepasst, dass sie konform zur Zeichencodierung der empfangenden Plattform ist. Wenn zum Beispiel Daten in U-Format in Little-Endian-Zeichencodierung auf einer Plattform mit Big-Endian-Zeichencodierung ankommt, werden die Daten in Big-Endian-Zeichencodierung umgewandelt, bevor sie an das Programm übergeben werden. Wenn diese Daten wieder zurückgesendet werden, dann werden sie wieder in Little-Endian-Zeichencodierung zurückgewandelt.

Logische Bedingungen

In einer logischen Bedingung können Unicode-Operanden zusammen mit alphanumerischen und binären Operanden benutzt werden. Wenn nicht alle Operanden Unicode-Operanden (Format U) sind, werden der zweite alle folgenden Operanden in das Format des ersten Operanden umgewandelt. Wenn ein binäre Operand (Format B) als zweiter oder nachfolgender Operand angegeben ist, muss die Länge des Operanden gerade sein. Es wird angenommen, dass der binäre Operand Unicode-Codepoints enthält.

Wenn der erste Operand ein Unicode-Operand ist (Format U) und der Vergleich daher als Unicode-Vergleich durchgeführt wird, wird der Unicode-Algorithmus benutzt. Der ICU-Algorithmus führt keinen einfachen Binärvergleich durch. Zum Beispiel:

- einige Codepoints, z.B. "U+0000" werden während des Vergleichsvorgangs ignoriert,
- zusammengesetzte Zeichen werden als gleich dem äquivalenten einzelnen Codepunkt angesehen (zum Beispiel der deutsche Buchstabe "ä", der durch "U+00E4" dargestellt wird, und die Kombination der Codepoints "U+0061" und "U+0308" werden von ICU als gleich betrachtet).

Anmerkung: Der Vergleich eines alphanumerischen und einen Unicode-Operanden kann, abhängig von der Reihenfolge der Felder, unterschiedliche Ergebnisse liefern.

Siehe auch *Logische Bedingungen* im *Leitfaden zur Programmierung*.

Systemvariablen

*CODEPAGE

Die Systemvariable `*CODEPAGE` gibt den IANA-Namen der Standard-Codepage zurück, die intern von Natural für Konvertierungen zwischen Unicode- und Codepage-Formaten benutzt wird.

*LOCALE

Die Systemvariable `*LOCALE` enthält die Sprache und das Land der aktuellen Locale.

Große und dynamische Variablen

Das Format kann bei großen und dynamischen Variablen benutzt werden. Bei dynamischen Variablen gibt die Systemvariable `*LENGTH` die Anzahl der UTF-16-Codeeinheiten zurück.

Siehe auch *Dynamische Variablen* im *Leitfaden zur Programmierung*.

Session-Parameter

Folgende Session-Parameter stehen zur Verfügung:

Parameter	Beschreibung
`DL`	Bestimmt die Ausgabelänge eines Felds im Format A oder U. Siehe auch *Ausgabelänge - der DL Parameter* im *Leitfaden zur Programmierung*.
`EMU`	Unicode-Editiermaske.
`ICU`	Unicode-Einfügungszeichen.
`LCU`	Vorangestellte Unicode-Zeichen.
`TCU`	Nachgezogene Zeichen in Unicode.

Session-Parameter DL im Vergleich zu AL

Solange wie Natural noch nicht Unicode-fähig war, war die Länge von alphanumerischen Feldern immer identisch mit der Anzahl der Spalten, die zum Anzeigen des Felds benötigt wurden (Anzahl der Anzeigespalten genannt). Das traf auch bei den ostasiatischen Sprachen zu, die DBCS-Codepages benutzen: Ein Feld im Format A kann nur die Hälfte der Zeichen aufnehmen (A10 zum Beispiel resultiert in A5).

Beispiel:

```
DEFINE DATA LOCAL
1  #A8 (A8)
END-DEFINE
#A8 := 'computer'
WRITE #A8
#A8 := '電腦系統'
WRITE #A8
END
```

Der oben gezeigte Code ergibt folgende Ausgabe:

```
Page      1 ...

computer
電腦系統
```

Bei Feldern im Format U ist die Länge eines Felds nicht mehr identisch mit der Anzahl der Spalten. Zeichen im Format U können eine schmale Länge (z.B. lateinische Zeichen), eine breite Länge (z.B. chinesische Zeichen) oder keine Länge (z.B. zusammengesetzte Zeichen) haben. Deshalb ist völlig unbekannt, wie viele Anzeigespalten ein Feld im Format U benötigt, denn es ist abhängig vom Inhalt des Feldes. Natural kann nicht automatisch entscheiden, wie viele Spalten im Bildschirm reserviert werden müssen: Wird die maximale Länge angenommen, dann werden in der lateinischen Zeichenausgabe große Lücken auftreten. Wird die minimale Länge angenommen, können chinesische Zeichenausgaben nicht vollständig angezeigt werden. Deshalb muss der Natural-Programmierer die Anzeigebreite eines Feldes definieren. Dazu dient der `DL`-Parameter. Der `AL`-Parameter kann für diesen Zweck nicht benutzt werden, weil er bewirkt, dass der Teil des Feldes, der die definierte Feldlänge übersteigt, abgeschnitten wird. Es sollen aber keine Zeichen von dem U-Format-Feld abgeschnitten werden. Es geht vielmehr darum, die Anfangsposition des nachfolgenden Feldes zu definieren.

Beispiel:

```
DEFINE DATA LOCAL
1  #U8 (U8)
1  #U4 (U4)
END-DEFINE
#U8 := 'computer'
WRITE #U8
#U4 := U'電腦系統'
WRITE #U4 (DL=8)
END
```

Der oben gezeigte Code ergibt die gleiche Ausgabe wie oben:

```
Page      1 ...

computer
電腦系統
```

Auf Windows ist es in einer Remote Development-Umgebung mit dem Natural Web I/O Interface Client möglich in einem Feld zu blättern, wenn der für den `DL`-Parameter angegebene Wert kleiner als die reale Anzeigebreite des Feldes ist.

EMU, ICU, LCU, TCU im Vergleich zu EM, IC, LC, TC

Die Parameter EMU, ICU, LCU und TCU ermöglichen es, Zeichen zu verwenden, die nicht in der Standard-Codepage enthalten sind. Sie werden innerhalb des Programms in Unicode-Format gespeichert. Diese Parameter können bei allen Feldformaten benutzt werden.

Die Parameter EM, IC, LC und TC können ebenfalls bei U-Format-Feldern benutzt werden. Diese Parameter können außerdem von Nutzen sein, wenn in der Standard-Codepage enthaltene Zeichen in anderen Codepages abweichende Zeichencodierungen haben. Das Euro-Zeichen (€) zum Beispiel hat den Codepoint "0x80" in der Codepage "windows-1252" (Latin 1), jedoch in der Codepage "windows-1251" (Cyrillic) den Codepoint "0x88".

Somit stellt die Benutzung eines Unicode-Parameters (EMU, ICU, LCU oder TCU) sicher, dass das Euro-Zeichen immer korrekt angezeigt wird, egal welche Codepage auf dem PC installiert ist.

Beispiel für EMU:

```
DEFINE DATA
LOCAL
 01 EMPLOYEES-VIEW VIEW OF EMPLOYEES
   02 FIRST-NAME
   02 NAME
   02 SALARY (1)
END-DEFINE
*
  READ (6) EMPLOYEES-VIEW
    DISPLAY NAME FIRST-NAME SALARY(1) (EMU=999,999€)
  END-READ
*
END
```

Der oben gezeigte Code ergibt folgende Ausgabe:

```
Page     1                                                    05-12-15  11:45:36

        NAME              FIRST-NAME       ANNUAL
                                           SALARY
-------------------- -------------------- --------

ADAM                 SIMONE               159,980€
MORENO               HUMBERTO             165,810€
BLOND                ALEXANDRE            172,000€
MAIZIERE             ELISABETH            166,900€
CAOUDAL              ALBERT               167,350€
VERDIE               BERNARD              170,100€
```

Beispiel-Programme

Die Library SYSEXPG enthält Beispiel-Programme für die Unicode- und Codepage-Unterstützung in Natural:

Programm	Beschreibung
UNICOX01	Listet alle Unicode-Zeichen auf.
UNICOX02	Konvertiert Unicode-Zeichen in Codepoints und umgekehrt.
CODEPX01	Listet alle Codepages auf und gibt an, ob die Codepage in Natural unterstützt wird und welche Zeichenkodierung sie verwendet. Bei allen unterstützen Codepages werden Services angeboten, die die Zeichen der Codepage auflisten und mit denen eine Zeichenkette von der Codepage in ihre hexadezimale Darstellung und umgekehrt konvertiert werden kann.
CODEPXL1	Listet alle Zeichen einer beliebigen Ein-Byte-Codepage auf.
CODEPXL2	Listet alle Zeichen einer beliebigen Zwei-Byte-Codepage auf.
CODEPXC1	Konvertiert eine Zeichenkette von einer beliebigen Codepage in ihre hexadezimale Darstellung und umgekehrt.

8 Behandlung von Unicode-Ein-/Ausgaben in Natural-Anwendungen

- Unicode-Daten anzeigen und eingeben 56
- Natural Web I/O Interface Client 57

Die Natural-Laufzeitumgebung unterstützt Unicode. Unicode-Zeichen werden in die Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) umgewandelt, bevor sie auf dem Terminal angezeigt werden. Unicode-Zeichen, für die es kein Äquivalent in der Standard-Codepage gibt, werden durch ein Ersatzzeichen ersetzt.

Mit dem Natural Web I/O Interface unter SPoD werden Unicode-Zeichen vollständig durch die Terminalemulation unterstützt. In diesem Fall werden Felder im Format U angezeigt und die Feldeingabe ist korrekt in Unicode möglich. Sie werden nicht in das Äquivalent in der Standard-Codepage umgewandelt. Das Natural Web I/O Interface wird durch den Konfigurationsparameter `TERMINAL_EMULATION=WEBIO` im Natural Development Server aktiviert. Die Systemvariable `*DEVICE` enthält dann den Wert `BROWSER`.

Unicode-Daten anzeigen und eingeben

Anmerkungen:

1. Unicode-Daten können nicht auf Terminals des Typs 3270 angezeigt werden.
2. Wenn Anwendungen auf Natural für Großrechner laufen, gilt das im Abschnitt ***Natural Web I/O Interface Client*** Gesagte.

Wenn Natural in einer Terminalemulation oder auf einem Terminal, z.B. IBM 3270/3279 läuft, wird die Seite in die Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) umgewandelt, bevor sie angezeigt wird. Dadurch werden alle Zeichen, die nicht in der Standard-Codepage enthalten sind, durch das Ersatzzeichen ersetzt. Gleichermaßen sind Eingaben nur im Codepage-Format möglich. Sie werden in Unicode-Format umgewandelt, bevor sie einem Feld im Format U zugewiesen werden. Dabei ist zu berücksichtigen, dass das Ersatzzeichen durch die ICU Conversion Tables definiert wird. Abhängig von diesem Zeichen ist es möglich, dass bei einer Terminalemulation Müll angezeigt wird. Es ist jedoch sehr zu empfehlen, das Natural Web I/O Interface zu benutzen, wenn Zeichen angezeigt werden, die nicht in der Standard-Codepage enthalten sind.

Bei Codepage-orientierten Großrechner-Terminals ist es wichtig, dass die passende Codepage gewählt wird. Die Standard-Codepage von Natural, die Codepage des Terminals und sogar die von dem Terminal verwendete Schriftart sind ausschlaggebend für die Befähigung des Terminals, bestimmte Zeichen korrekt anzuzeigen.

Natural Web I/O Interface Client

Durch das Natural Web I/O Interface wird zur Laufzeit die Ein- und Ausgabe in Unicode vollständig unterstützt. Wenn eine Anwendung in der Terminalemulation läuft und Zeichenketten angezeigt werden, kann es sein, dass manche Unicode-Zeichen nicht korrekt angezeigt werden.

Der Natural Web I/O Interface Client dient dazu, Nicht-GUI-Informationen anzuzeigen, die Unicode-Zeichen enthalten. Der Client kann in den nachfolgend beschriebenen Umgebungen benutzt werden.

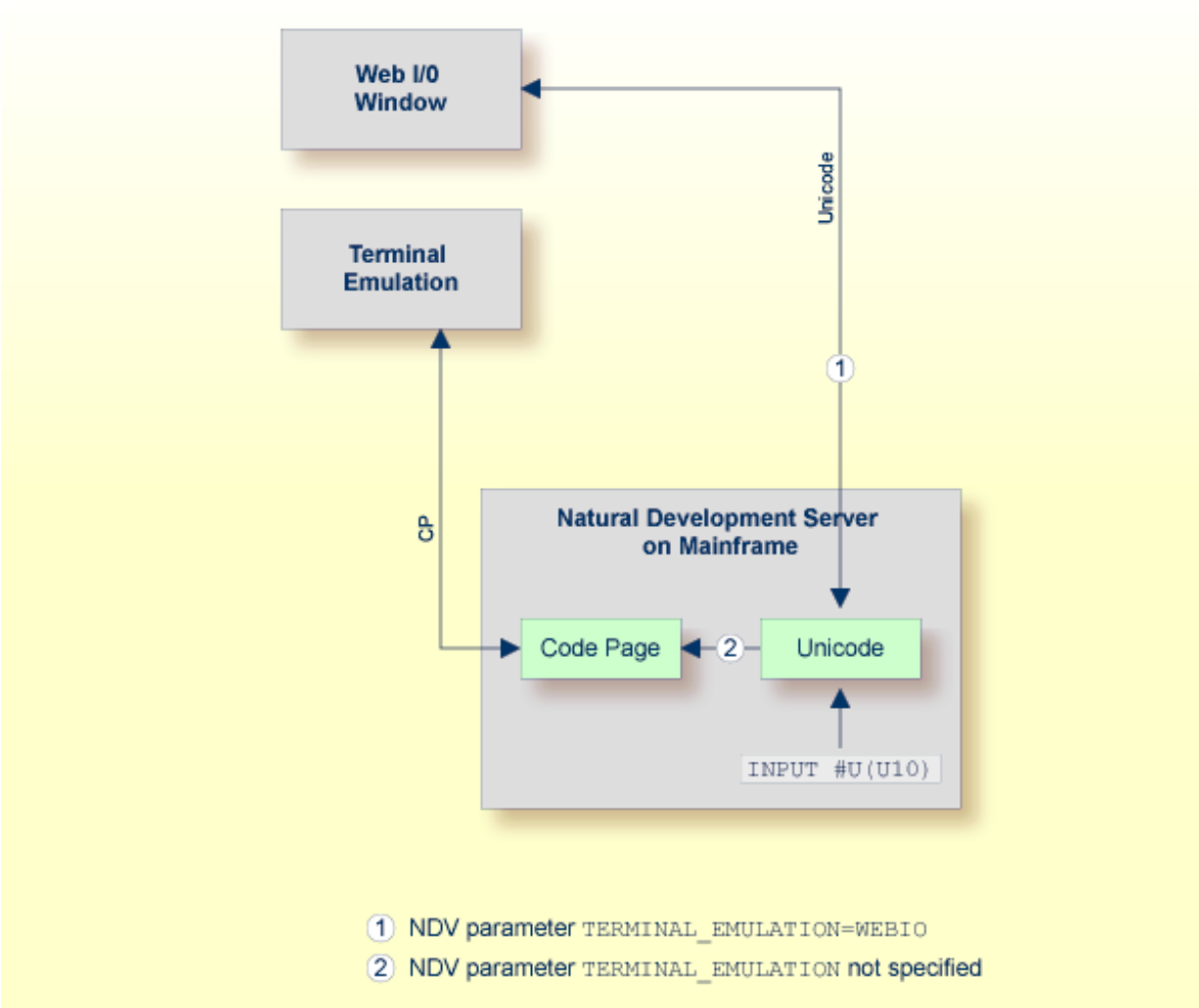
- SPoD-Umgebung
- Laufzeit-Umgebung

SPoD-Umgebung

Der Natural Web I/O Interface Client kann aufgerufen werden, wenn Sie Natural for Windows benutzen und mit Natural Studio in einer Remote Development-Umgebung (SPoD) arbeiten, siehe *Natural Web I/O Interface Client* in *Remote Development Using SPoD* (Teil der *Natural for Windows*-Dokumentation).

Wenn der Natural Web I/O Interface Client benutzt wird, erscheint das Web-Ein-/Ausgabefenster anstelle des Terminalemulationsfensters, das bei Großrechner-Umgebungen benutzt wird und nicht Unicode-fähig ist.

Die folgende Grafik zeigt die SPoD-Umgebung für Unicode-Anwendungen mit dem Natural Development Servers (NDV) auf Großrechnern:



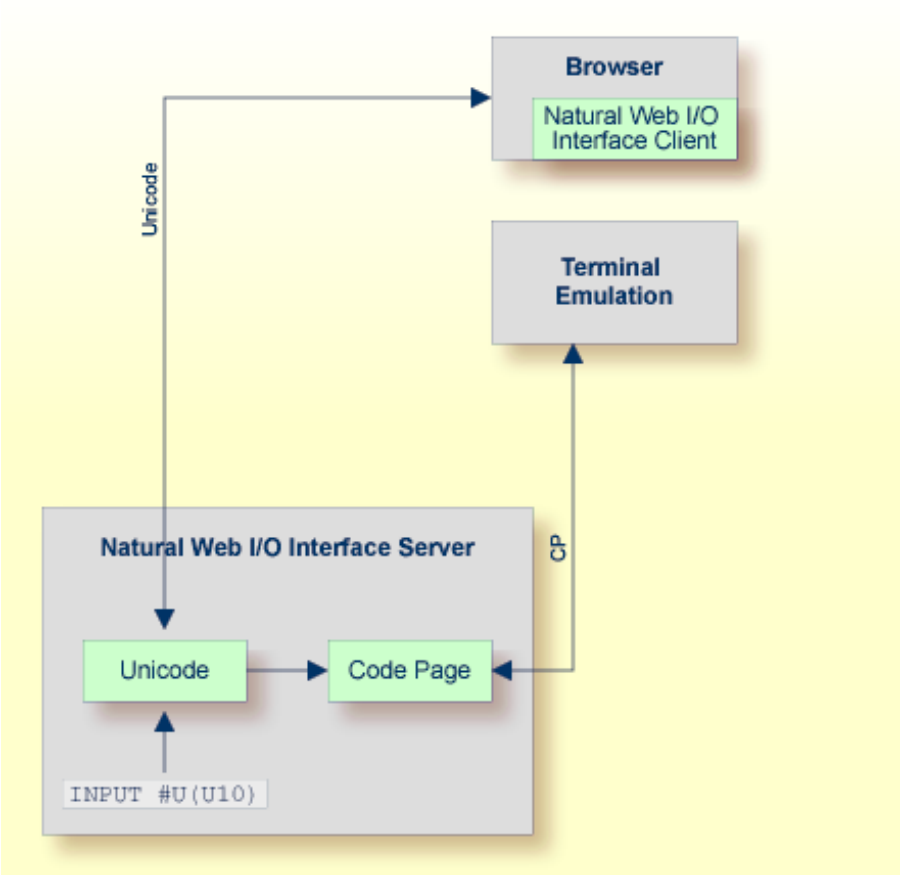
Damit der Natural Web I/O Interface Client aufgerufen werden kann, muss der Natural Development Server folgendermaßen konfiguriert werden:

- Wenn Sie den Natural Web I/O Interface Client in einer Remote-Großrechner-Umgebung benutzen wollen, muss der NDV-Konfigurationsparameter `TERMINAL_EMULATION` beim NDV-Server auf `WEBIO` gesetzt werden. Siehe *NDV Configuration Parameters* in der *Natural Development Server*-Dokumentation.
- Der Natural-Profilparameter `TMODEL` (IBM 3270-Terminal-Modell) kann benutzt werden, um die Größe des Benutzerbildschirms festzulegen.

Laufzeit-Umgebung

Der Natural Web I/O Interface Client erscheint, wenn Anwendungen mit Natural laufen. Der Client läuft in einem Web/Anwendungs-Server.

Die folgende Grafik zeigt die Laufzeitumgebung bei Unicode-Anwendungen:



Natural erkennt automatisch, ob die Session aus dem Natural Web I/O Interface Client oder aus einer Terminalemulation gestartet worden ist.

Dazu muss der Natural Web I/O Interface Server installiert und konfiguriert worden sein. Siehe *Natural Web I/O Interface*. Außerdem muss das Web I/O Terminal Converter-Modul `NATWEB` mit dem Natural-Nukleus verlinkt sein. Der Natural-Profilparameter `TMODEL` (IBM 3270-Terminal-Modell) kann benutzt werden, um die Größe des Benutzerbildschirms festzulegen.

9 Unterstützung bidirektionaler Sprachen

- Allgemeines zu bidirektionalen Sprachen 62
- Zeichenlaufrichtung im Bildschirm 62
- Zeichenlaufrichtung im Feld 63
- Arabische Formgebung 64

Allgemeines zu bidirektionalen Sprachen

Einige Sprachen, z. B. Arabisch und Hebräisch, werden von rechts nach links (rechtsläufig, right-to-left, RTL) geschrieben, während die meisten Sprachen, z. B. Englisch und Deutsch, von links nach rechts geschrieben werden (linksläufig, left-to-right, LTR). Text, der Zeichen sowohl in rechtsläufiger als auch linksläufiger Schreibweise enthält, wird als bidirektionaler Text bezeichnet.

Die Unterstützung bidirektionaler Sprachen wird nicht automatisch aktiviert. Der Benutzer muss dazu immer alle erforderlichen Parameter (z.B. `PM=I`) so wie nachfolgend beschrieben angeben

Die Ausgabe von Natural-Programmen kann durch Angabe des Profilparameters `PM` (Druck-/Anzeige-Modus), des Terminalkommandos `%V` (Steuerung des Print-Modus) und des Session-Parameters `PM` gesteuert werden.

Zusätzlich wird der Profilparameter `DO` (Anzeige-Reihenfolge von Ausgabedaten) benutzt, um Anwendungen zu unterstützen, die ursprünglich für Terminals geschrieben wurden, die inversen Print-Modus (von rechts nach links), jedoch keine bidirektionalen Daten unterstützen. Diese Anwendungen erzeugen die Anzeigereihenfolge von bidirektionalen Daten innerhalb des Codes der Anwendung. Mit dem Profilparameter `DO` werden diese Anwendungen in die Lage versetzt, auch mit Eingabe-/Ausgabegeräten kompatibel zu laufen, die bidirektionale Daten unterstützen. Dies ist beispielsweise der Fall, wenn eine Anwendung in einem Browser mit dem Natural Web I/O Interface läuft.

Zeichenlaufrichtung im Bildschirm

Der Profilparameter `PM` definiert die Standard-Laufrichtung der Schriftzeichen im Bildschirm. Wenn `PM` auf `R` (Reset) gesetzt ist, ist die Standard-Laufrichtung linksläufig (von links nach rechts). Wenn `PM` auf `I` (Invers), gesetzt ist, ist die Standard-Laufrichtung rechtsläufig (von rechts nach links). Alle nicht-alphanumerischen Felder, Systemvariablen und PF-Tastenzeilen werden durch Natural automatisch invertiert, damit sie korrekt von rechts nach links laufend angezeigt werden, wenn die Laufrichtung im Bildschirm rechtsläufig ist.

Das Terminalkommando `%V` kann benutzt werden, um die Laufrichtung der Schriftzeichen im Bildschirm zu ändern. Laufrichtung im Bildschirm von rechts nach links verläuft, wird das Layout des aktuellen Fensters gespiegelt. was bedeutet, das der Ausgangspunkt aller Fensterbestandteile oder Felder in der rechten oberen Ecke des Bildschirms liegt. Durch Eingabe von `%VON` wird die Laufrichtung im Bildschirm in rechtsläufig geändert. Durch Eingabe von `%VOFF` wird sie umgekehrt und in linksläufig geändert.

Zeichenlaufrichtung im Feld

Der Session-Parameter PM dient zur Umkehr der Zeichenlaufrichtung in einem Feld. Die Wirkung, die mit der „Umkehr der Zeichenlaufrichtung“ in einem Feld erzielt wird, ist abhängig von dem Statement, in dem der Session-Parameter PM benutzt wird und von der Plattform. Wird der Parameter PM in einem MOVE-Statement benutzt, wird der Inhalt des Feldes einfach nur umgekehrt (d.h. das erste Zeichen wird zum letzten Zeichen usw.). Das Ergebnis ist nicht abhängig von den Zeichen im Feld. Leerzeichen am Ende werden entfernt, bevor die Zeichenlaufrichtung im Feld umgekehrt wird.

Beispiel: Das folgende Programm

```
DEFINE DATA LOCAL
1  TEST1  (A10)
1  TEST2  (A10)
END-DEFINE
TEST1 := 'program'

MOVE TEST1 (PM=I) TO TEST2
INPUT TEST1 (AD=O) TEST2 (AD=O)

END
```

erzeugt folgende Ausgabe:

```
TEST1 program    TEST2 margorp
```

wobei "margorp" die umgekehrte Version von "program" ist.

Wenn der Parameter PM bei IO-Statements wie INPUT oder DISPLAY verwendet wird, ist seine Wirkung noch komplexer. In diesem Fall richtet sich die Feldrichtung nach der Richtung des Bildschirms:

- Wenn die Richtung des Bildschirms von links nach rechts verläuft und PM=I auf ein Feld angewendet wird, ändert sich die Feldrichtung in rechts nach links.
- Wenn die Richtung des Bildschirms von rechts nach links verläuft und PM=I auf ein Feld angewendet wird, ändert sich die Richtung des Feldes in links nach rechts.

Auf Browser-Terminals (Natural Web I/O Interface) bedeutet „Umkehrung der Feldrichtung“ nicht, dass die Zeichen des Feldes einfach umgedreht werden. Stattdessen wird der komplexe bidirektionale Algorithmus angewendet. Auf zeichenorientierten Terminals hingegen werden die Zeichen eines Feldes nicht umgeordnet, sondern einfach umgekehrt.

Im folgenden Beispiel wurden die Zeichen, die der Variablen TEST zugewiesen sind, in der folgenden Reihenfolge eingegeben:

```
a b c   ש נ ב   1 2 3
```

Wenn die Zeichen in der oben beschriebenen Reihenfolge eingegeben werden, wird das Programm wie folgt angezeigt, da die Zeichen einfach in der Reihenfolge der Tastatureingabe angezeigt werden.

```
DEFINE DATA LOCAL
1  TEST  (A20)
END-DEFINE
TEST := 'abc  בנש  123'

SET CONTROL 'voff'

INPUT TEST (AD=O)  /
      TEST (AD=O PM=I)

SET CONTROL 'von'

INPUT TEST (AD=O)  /
      TEST (AD=O PM=I)
END
```

Dieses Programm erzeugt zwei identische Bildschirme, weil die Statements `SET CONTROL 'voff'` und `SET CONTROL 'von'` nicht für alphanumerische Felder gelten. Beide Bildschirme sehen wie folgt aus:

```
TEST abc  בנש  123
TEST          321  שנב  cba
```

Arabische Formgebung

Im arabischen Text sind normalerweise alle Zeichen einer Zeichenkette miteinander verbunden. Aus diesem Grund haben arabische Zeichen bis zu vier Darstellungsformen: die isolierte, die finale, die initiale und die mediale Form. Welche Form verwendet wird, hängt von der Position des Zeichens in der Zeichenkette ab. Zum Beispiel hat das arabische Zeichen "MEEM" die folgenden Formen in Unicode:

U+0645	م	ARABIC LETTER MEEM
U+FEE1	م	ARABIC LETTER MEEM ISOLATED FORM
U+FEE2	م	ARABIC LETTER MEEM FINAL FORM
U+FEE3	ـمـ	ARABIC LETTER MEEM INITIAL FORM
U+FEE4	ـمـ	ARABIC LETTER MEEM MEDIAL FORM

Darüber hinaus werden Zeichen zu einer neuen Form zusammengesetzt, wenn sie nacheinander in einer Zeichenkette erscheinen. Das wird als „Ligatur“ bezeichnet. Beispiel: Die Zeichen

U+0644	ل	ARABIC LETTER LAM
U+0627	ا	ARABIC LETTER ALEF

haben die folgende zusammengesetzte Form:

U+FEFB	ﻻ	ARABIC LIGATURE LAM WITH ALEF ISOLATED FORM

Unicode-Zeichenketten sollten nur die arabischen Zeichen des arabischen Blocks (U+0600 bis U+06FF) oder des arabischen Ergänzungsblocks (U+0750 bis U+077F) enthalten. Es wird nicht empfohlen, die Darstellungsformen in normalem arabischen Text zu verwenden. Es ist Aufgabe der Benutzeroberfläche, die korrekten Formen der Zeichen anzuzeigen.

„Geformt“ (Shaped) bedeutet, dass jedes arabische Basiszeichen in die entsprechende arabische Darstellungsform umgewandelt wird. Die Zeichenfolge kann jede der vier Darstellungsformen eines Zeichens enthalten. Wenn zum Beispiel das Zeichen (Buchstabe) U+0645 (ARABIC LETTER MEEM) als letztes Zeichen einer Zeichenfolge verwendet wird, wird es in U+FEE2 (ARABIC LETTER MEEM FINAL FORM) umgewandelt.

"Ungeformt" (Unshaped) bedeutet, dass jedes Zeichen nur durch seine Grundform dargestellt wird. Zum Beispiel wird anstelle von U+FEE2 (ARABIC LETTER MEEM FINAL FORM) U+0645 (ARABIC LETTER MEEM) verwendet. Die Umwandlung in die richtige Darstellungsform wird von der Rendering-Engine des Ausgabegeräts vorgenommen.

Natural-Zeichenketten werden intern als ungeformte Alpha- oder Unicode-Zeichenketten dargestellt. Werden Zeichenketten mit einem Browser unter Verwendung des Natural Web I/O Interface-Clients oder des `PROCESS PAGE`-Statements angezeigt, ist keine Umwandlung erforderlich, da die Rendering-Engine des Browsers für die korrekte Darstellung sorgt. Eingehende Zeichenketten von solchen Geräten sind bereits ungeformt und können direkt an Natural übergeben werden. Wird eine Zeichenkette auf einem Terminal wie 3279 oder einem Terminalemulator wie IBM Personal Communications angezeigt, muss sie in die geformte Form umgewandelt werden, da das Terminal selbst nicht für die korrekte Darstellung sorgt. Dementsprechend liegen eingehende Zeichenketten in der geformten Form vor und müssen in die ungeformte Form umgewandelt werden, damit sie von Natural korrekt verarbeitet werden können. Die am meisten verbreitete Codepage für arabische Terminals auf dem Großrechner ist IBM420. Im Vergleich zu Unicode ist die Anzahl der Zeichen reduziert und nicht jede Form eines Zeichens ist enthalten. Bei der Umwandlung von Zeichenketten in IBM420 werden nicht verfügbare Formen eines Zeichens durch eine ähnliche Darstellungsform ersetzt. Zum Beispiel wird die mediale Form des arabischen Buchstabens MEEM (U+FEE4) durch die Anfangsform (U+FEE3) des Zeichens ersetzt.

In der arabischen EBCDIC-Codepage IBM420 wird das arabische Zeichen „MEEM“ durch die folgenden Darstellungsformen dargestellt:

H'BA'	م	ARABIC LETTER MEEM
H'BB'	ـمـ	ARABIC LETTER MEEM INITIAL FORM

Arabisches Endfragment

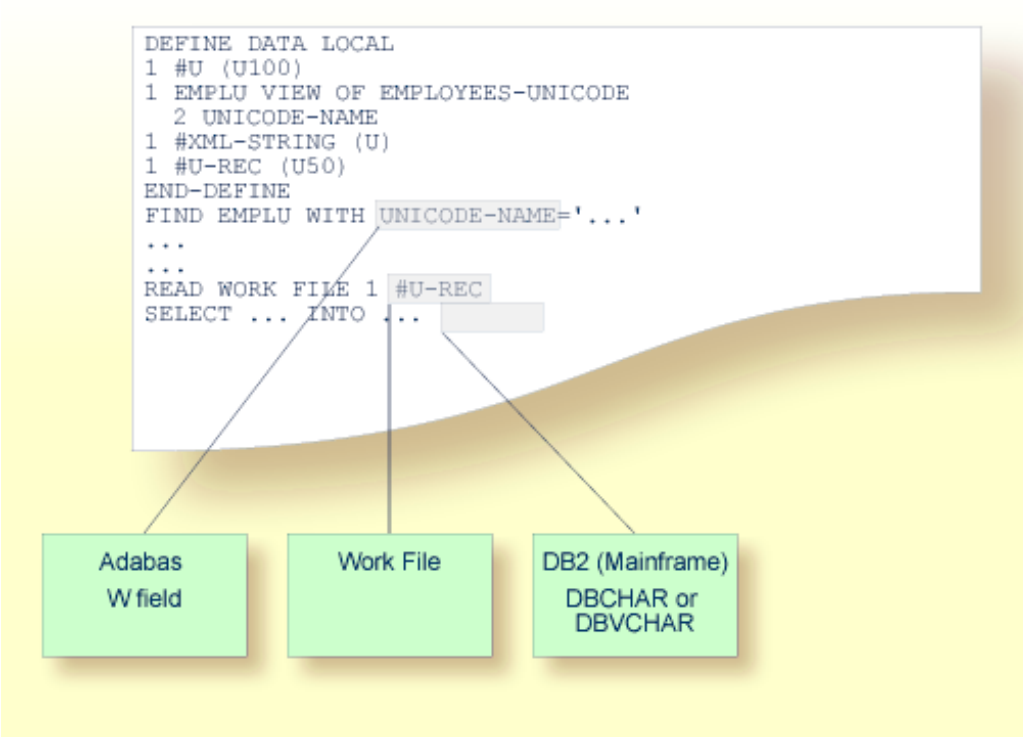
Die arabischen Zeichen SEEN (U+0633), SHEEN (U+0634), SAD (U+0635) und DAD (U+0636) (Seen-Familie) werden auf Terminals als zwei Bytes angezeigt, wenn sie in der Endform erscheinen. Die Codepage IBM420 enthält ein so genanntes „arabisches Endfragment", das die Endform eines Zeichens der Seen-Familie auf Terminals oder Terminalemulatoren vervollständigt. Natürlich benötigt das arabische Endfragment eine zusätzliche Position auf dem Bildschirm. Das arabische Endfragment wird von den Browsern nicht benötigt. Wird eine Zeichenkette mit der Endform eines Zeichens der Seen-Familie in einen Browser eingegeben (Natural Web I/O Interface-Client oder `PROCESS PAGE`-Statement) und anschließend auf einem Terminal angezeigt, wird das arabische Endfragment an die Zeichenkette angehängt, was zur Folge hat, dass die Länge der Zeichenkette zunimmt. Wird eine Zeichenkette mit der Endform eines Zeichens der Seen-Familie über ein Terminal oder einen Terminalemulator eingegeben und anschließend in einem Browser angezeigt, wird das arabische Endfragment aus der Zeichenkette entfernt.

10 Unicode-Datenspeicherung

- Zugriff auf Unicode-Daten und Parameter 68
- Datenbankmanagementsystem-Schnittstellen 68
- Arbeitsdateien und Druckdateien 69

Zugriff auf Unicode-Daten und Parameter

Die folgende Grafik veranschaulicht, wie auf Unicode-Daten und -Parameter zugegriffen wird:



Datenbankmanagementsystem-Schnittstellen

Folgende Themen werden behandelt:

- Auf Unicode-Daten in einer Adabas-Datenbank zugreifen

- Zugriff auf Unicode-Daten in einer DB2-Datenbank

Auf Unicode-Daten in einer Adabas-Datenbank zugreifen

Natural ermöglicht es Benutzern, auf Wide-Character-Fields (Format W) in einer Adabas-Datenbank zuzugreifen.

Datendefinitionsmodule
Adabas Wide-Character-Fields (W) werden auf Natural-Format U (Unicode) abgebildet.

Zugriffskonfiguration
Natural erhält Daten aus Adabas und sendet Daten zurück an Adabas und benutzt dazu UTF-16 als übliche Zeichencodierung.

Diese Kodierung wird mit dem Profilparameter `OPRB` angegeben und an Adabas mit der `Open`-Anforderung gesendet. Sie wird für Wide-Character-Fields benutzt und gilt für die gesamte Adabas-Benutzer-Session.

Weitere Informationen siehe *Unicode-Daten* im Abschnitt *Daten in einer Adabas-Datenbank aufrufen* im *Leitfaden zur Programmierung*.

Zugriff auf Unicode-Daten in einer DB2-Datenbank

Natural ermöglicht es, dass Benutzer auf `CHAR`- und/oder `WCHAR`-Felder in einer DB2-Datenbank als Unicode-Daten zuzugreifen.

Siehe auch *Natural for DB2* in der *Datenbankmanagementsystem-Schnittstellen*-Dokumentation.

Arbeitsdateien und Druckdateien

Folgende Themen werden behandelt:

- Arbeitsdateien (Work Files)

- Druckdateien (Print Files)

Arbeitsdateien (Work Files)

Beim Schreiben oder Lesen von Arbeitsdateien werden Unicode-Daten nicht besonders berücksichtigt. Wie alle anderen Daten werden Unicode-Daten so geschrieben und gelesen, wie sie sind, ohne Umwandlung.

Druckdateien (Print Files)

Wenn Unicode-Daten an Druckdateien gesendet werden, finden eine oder zwei Umsetzungen statt.

Als ersten Schritt werden in einer Druckzeile enthaltene Unicode-Daten in die Standard-Codepage der Session umgewandelt. Infolgedessen werden alle Zeichen, die nicht in dieser Standard-Codepage enthalten sind, durch das Ersatzzeichen ersetzt.

Bevor die umgewandelte Druckzeile an die eigentliche Druckzugriffsmethode übergeben wird, erfolgt eine zusätzliche Prüfung, ob eine Codepage für den logischen Drucker angegeben worden ist. Das kann mit dem `CODEPAGE`-Operanden des `DEFINE PRINTER`-Statements oder dem Schlüsselwort-Subparameter `CP` des Profilparameters `PRINT` vorgenommen werden. Wenn eine solche Codepage angeben worden ist, dann wird im zweiten Schritt die gesamte Druckzeile (nicht nur ihr Unicode-Teil) entsprechend umgewandelt.

Die umgewandelte Druckzeile wird an die Zugriffsmethode übergeben, was bedeutet, dass Druckzugriffsmethoden *keine* Unicode-Daten erhalten.

Beispiel:

```
DEFINE PRINTER (1) CODEPAGE 'IBM01140'
WRITE (1)   'HELLO'    U'WORLD'
END
```

11 Migration existierender Anwendungen

- Einfluss von Unicode auf existierende Anwendungen 72
- Existierende Objekte migrieren 72
- Existierende Anwendungen auf Unicode-Unterstützung erweitern 74
- Migration von Natural Remote Procedure Calls (RPC) 74

Einfluss von Unicode auf existierende Anwendungen

Es gibt keinen Einfluss von Unicode auf existierende Anwendungen. Das bedeutet, das existierende Natural-Anwendungen ohne Änderungen ausgeführt werden sollten. Vergewissern Sie sich, dass die Profilparameter `CFICU` und `CP` auf `OFF` gesetzt sind. In diesem Fall ist es nicht nötig, irgendwelche Komponenten zu installieren, die mit den ICU for Adabas & Natural (ICS) geliefert werden. Es wurden lediglich die Ein-/Ausgabepuffer deutlich vergrößert, da die Attribute verbessert wurden, um potenzielle Unicode-Felder zu unterstützen. Wenn `CP`auf `OFF` gesetzt ist, wird der Inhalt der Systemvariablen `*CODEPAGE` gelöscht und die vertrauten Umsetzungstabellen (z.B. Standard-Tabelle und alternative Tabelle) werden weiterhin für Ein-/Ausgabe-Umsetzungen benutzt.

Existierende Objekte migrieren

Natural wurde so erweitert, dass die Codepage-Informationen auf mehreren Ebenen definiert werden können:

- Der Natural-Profilparameter `CP` definiert die Natural-Standard-Codepage.
- Bei manchen Objekten (Natural-Quellcode-Objekt, Natural-Batch-Ein-/Ausgabedateien, Print Reports, Adabas-Dateien) kann eine objektspezifische Codepage definiert werden.

Wenn weder eine objektspezifische Codepage noch eine Standard-Codepage definiert ist (d.h., es gilt die Einstellung `CP=OFF`), führt Natural keine Umwandlung von Daten durch.

Da es nicht möglich ist, die korrekte Codepage automatisch zu identifizieren, ist es wichtig, dass Sie die erforderliche Codepage selbst angeben. Dabei sind folgende Szenarien möglich:

Status	Aufwand	Maßnahme
Alle Daten liegen in der Codepage des Betriebssystems vor.	Keiner.	Keine.
Alle Daten werden mit einer Codepage gespeichert, aber diese Codepage unterscheidet sich von der Codepage des Betriebssystems.	Einfach.	Im Natural-Profilparameter `CP` muss die korrekte Codepage angegeben sein. Vergewissern Sie sich, dass das Ein-/Ausgabegerät diese Codepage unterstützt. Bei der Einstellung `CP=AUTO` läuft Natural zwangsweise mit der Codepage des Ein-/Ausgabegeräts.

Status	Aufwand	Maßnahme
Die Daten liegen in unterschiedlichen Codepages vor.	Abhängig von der Anzahl der Codesources und Codepages.	Die korrekte Codepage muss bei jedem Natural-Objekt angegeben werden: ■ **Sources** Speichern Sie jedes Objekt in der Session mit der korrekten Codepage. ■ **Batch Files** Setzen Sie im Natural-Profilparameter `CPOBJIN`, `CPSYNIN` und `CPPRINT` die korrekte Codepage. ■ **Adabas Files (ECS-fähig)** Setzen Sie im Natural-Profilparameter `OPRB` die Option `ACODE`.
Verschiedene Codepages sind in einem Objekt gemischt vorhanden (zum Beispiel in einer Source).	Hoch	Das Objekt muss im entsprechenden Codepage-Format neu geschrieben werden.

Quellcode-Objekte, die mit früheren Natural-Versionen gespeichert (`SAVE`) oder gespeichert und katalogisiert (`STOW`) worden sind, haben keine Codepage-Informationen. Das Codepage-Feld des Verzeichnisse (Directory) ist leer.

Da Natural-Quellcode-Objekte nicht in Unicode-Format gespeichert werden, muss das Quellcode-Objekt in die Standard-Codepage (Wert der Systemvariablen `*CODEPAGE`) umgesetzt werden, die für die Session gilt. Wenn die Codepage-Unterstützung abgeschaltet ist (`CP=OFF`), wird die Codepage-Information des Quellcode-Objekts ignoriert und es erfolgt keine Umwandlung. Alphanumerische Konstanten müssen an die Standard-Codepage angepasst werden, wenn sie in den Quellcode-Bereich geladen werden.

Da Natural-Quellcode-Objekte nicht in Unicode-Format gespeichert werden, müssen alphanumerische Konstanten beim Start des Objekts an die Standard-Codepage angepasst werden. Dies kann mit der Compiler-Option `CPAGE` erreicht werden. Wenn `CPAGE` auf `ON` gesetzt ist, wird eine zusätzliche Tabelle in das Objekt generiert. Der Natural Loader benutzt diese Tabelle, um jede alphanumerische Konstante in die Standard-Codepage umzusetzen (Wert der Systemvariablen `*CODEPAGE`). Je nach Anzahl an alphanumerischen Konstanten erhöht die zusätzliche Tabelle die Größe des resultierenden Objekts und die Umwandlung verbraucht zusätzliche CPU-Zeit.

Es ist wichtig, dass abhängige Objekte (z.B. ein Programm und ein lokaler Datenbereich (LDA), der von dem Programm benutzt wird) die gleiche Codepage benutzen. Wenn abhängige Objekte unterschiedliche Codepages benutzen, sollte sichergestellt werden, dass die verwendeten Zeichen (z.B. "#") in den benutzten Codepages auf dieselben Codepoints abgebildet werden. Die folgenden Objekte haben keine zugehörige objektspezifische oder datenspezifische Codepage:

- Datendefinitionsmodule (DDMs),
- Predict Rules,
- Predict XRef Data.

Es ist Vorsicht geboten, wenn solche Daten in Objekten benutzt oder von Objekten erzeugt werden, für die eine spezifische Codepage angegeben worden ist. Wenn die Anwendung selbst nicht notwendigerweise Codepage-fähig sein muss, und wenn Sie wollen, dass die Anwendung in Hinblick auf zu verarbeitende Daten Codepage-sensibel ist, sollten Sie den Wert `(ON,EXCEPTNEW)` beim Profilparameter `SRETAIN` in Betracht ziehen.

Existierende Anwendungen auf Unicode-Unterstützung erweitern

Existierende Anwendungen lassen sich leicht mit neuem, auf dem Format U basierendem Quellcode auf Unicode-Unterstützung erweitern. Für das Format U müssen (im Vergleich zum Format A) die folgenden Regeln berücksichtigt werden:

- Eine Redefinition des Formats U in ein Format ungleich U unter Benutzung des `REDEFINE`-Statements sollte vermieden werden, weil es zu gespaltenen Zeichen führen kann.
- Das Format U ist Endian-abhängig. Dies muss beim Übertragen (`MOVE`) zwischen den Formaten B und U berücksichtigt werden.
- Denken Sie daran, dass beim Übertragen (`MOVE`) von U nach A Daten verloren gehen können.

Wenn Sie das Format existierender Felder von A nach U ändern wollen, müssen Sie folgenden Regeln beachten:

- Code, der eine spezifische Zeichencodierung von Zeichenketten voraussetzt, muss geändert werden (zum Beispiel durch Vergleich mit einem Feld im Format B).
- Alle `REDEFINE`-Statements des Feldes müssen auf Gültigkeit geprüft werden.
- Ein `REDEFINE` nach N ist nicht möglich (d.h., Sie erhalten nicht das erwartete Ergebnis).
- Das Datenbankfeld muss nach Unicode migriert werden (vorausgesetzt, das wird von Ihrer Datenbank unterstützt).
- Es ist möglich, dass Sie die Länge des Feldes ändern müssen: Wenn ein Feld im Format A BDCS-Zeichen enthält, dann ist beim Feld im Format U nur die halbe Länge erforderlich.

Migration von Natural Remote Procedure Calls (RPC)

Der Profilparameter `CP` wird in Verbindung mit dem Parameter-Makro `NTCPAGE` (im Source-Modul `NATCONFG`) benutzt, um den Namen der Standard-Codepage für Natural-Daten anzugeben oder automatisch den Codepage-Namen vom Benutzer-Terminal zu nehmen.

Der Schlüsselwort-Subparameter `CPRPC` wird beim Profilparameter `RPC` und dem entsprechenden Makro `NTRPC` benutzt.

12 Häufig gestellte Fragen

- Warum erhalte ich beim Start den Fehler 'Invalid code page specified'? 76
- Was ist die Standard-Codepage? 76
- Welche Standard-Codepage wird benutzt? 76
- Wie kann ich alle relevanten Natural-Codepage-Einstellungen anzeigen? 76
- Wie kann ich UTF-8-Zeichencodierung mit Natural-Code behandeln? 76
- Warum werden manche Zeichen nicht korrekt angezeigt? 77
- Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt bearbeiten möchte? 77
- Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt speichern möchte? 77
- Wie kann ich die Zeichencodierung eines Natural-Quellcodes herausfinden? 78
- Wie kann ich die Zeichencodierung eines Natural-Quellcodes ändern? 78
- Welches Ersatzzeichen wird verwendet, wenn ein Zeichen nicht konvertiert werden kann? 78
- Kann ich Natural-Quellcodes mit früheren Natural-Versionen benutzen, die noch keine Codepage-Unterstützung bieten? 78

Warum erhalte ich beim Start den Fehler 'Invalid code page specified'?

Die Codepage, die Sie mit dem Profilparameter `CP` definiert haben, existiert entweder nicht (gültige Codepages siehe ***http://demo.icu-project.org/icu-bin/convexp*** und zugehörige IANA-Namen siehe ***http://www.iana.org/assignments/character-sets***) oder ist eine für die Plattform ungültige Standard-Codepage (z.B. eine ASCII-Codepage kann nicht auf einer Großrechner-Plattform benutzt werden).

Prüfen Sie, ob gleicher IANA-Name, CCSID/CCSN oder Aliasname gemäß Angabe in `NATCONFG` benutzt wird.

Was ist die Standard-Codepage?

Die Standard-Codepage ist die Codepage, die das Ergebnis der Auswertung des Profilparameters `CP` ist.

Welche Standard-Codepage wird benutzt?

Die Standard-Codepage, die von Natural zur Konvertierung zwischen Codepage und Unicode und umgekehrt benutzt wird, kann durch Anzeige des Inhalts der Systemvariablen `*CODEPAGE` festgestellt werden.

Wie kann ich alle relevanten Natural-Codepage-Einstellungen anzeigen?

Benutzen Sie das Systemkommando `CPINFO`.

Wie kann ich UTF-8-Zeichencodierung mit Natural-Code behandeln?

Benutzen Sie das `MOVE ENCODED`-Statement für die Konvertierung von UTF-8 nach UTF-16: Die Codepage "UTF-8" muss für die A-Format-Variable benutzt werden.

Warum werden manche Zeichen nicht korrekt angezeigt?

Prüfen Sie, ob Sie die korrekte Codepage benutzen. Falls die Codepage korrekt ist, prüfen Sie, ob die gewählte Schriftart (Font) die Zeichen unterstützt, die Sie anzeigen wollen.

Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt bearbeiten möchte?

Das Quellcode-Objekt wird mit der zum Zeitpunkt der Erstellung gültigen Codepage gespeichert. Sie erhalten einen Konvertierungsfehler, wenn der Quellcode nicht von der Codepage des gespeicherten Quellcode-Objekts in die Codepage der aktuellen Natural-Session konvertiert werden kann. Sie können Natural mit der Codepage des Quellcode-Objekts starten, um eine Konvertierung zu vermeiden, oder Sie können nicht konvertierbare Zeichen in dem Fenster, das beim Start des Editors erscheint, anpassen.

Warum erhalte ich einen Fehler, wenn ich ein Natural-Quellcode-Objekt speichern möchte?

Wenn Sie über den Natural Single Point of Development (SPoD) mit einer Großrechnerumgebung verbunden sind, wird der Quellcode vom Großrechner in der SPoD-Umgebung in Unicode konvertiert und bearbeitet. Wenn er gespeichert wird, muss er in die Codepage des Natural-Servers konvertiert werden. Ein Konvertierungsfehler kann auftreten, wenn ein Unicode-Zeichen nicht in der Codepage der Natural-Server-Session abgebildet ist.

Wenn Sie sich in einer nativen Natural für Großrechner-Umgebung (ohne SPoD) befinden, erhalten Sie beim Speichern eines Quellcode-Objekts keine Fehlermeldungen, da ja keine Konvertierung erfolgt. Der Quellcode wird mit der Codepage-Information der aktuellen Natural-Session gespeichert.

Wie kann ich die Zeichencodierung eines Natural-Quellcodes herausfinden?

Codepage-Informationen sind Bestandteil des Natural Source Directory. Benutzen Sie das Systemkommando `LIST DIR`, um das Verzeichnis anzuzeigen.

Wie kann ich die Zeichencodierung eines Natural-Quellcodes ändern?

Sie sollten Ihre Natural-Session mit der gewünschten Codepage starten und den Natural-Profilparameter `CP` (Name der Standard-Codepage) benutzen. Setzen Sie den Natural-Profilparameter `SRETAIN` (Source-Format beibehalten) auf `OFF`. Bearbeiten Sie den Quellcode und speichern Sie ihn. Der Quellcode hat jetzt die geänderte Codepage-Information. Alternativ können Sie die Utility `SYSCP` benutzen, um die Codepage-Zuweisung eines Quellcodes zu prüfen oder zu ändern.

Welches Ersatzzeichen wird verwendet, wenn ein Zeichen nicht konvertiert werden kann?

Es wird das Ersatzzeichen der Codepage oder, falls in der Konfigurationsdatei angegeben, das Platzhalterzeichen verwendet.

Kann ich Natural-Quellcodes mit früheren Natural-Versionen benutzen, die noch keine Codepage-Unterstützung bieten?

Sie können auch mit früheren Natural-Versionen, die noch keine Codepage-Unterstützung bieten, auf Quellcodes zuzugreifen, die mit Codepage-Informationen gespeichert wurden. Das Layout des Quellcodes wurde nicht geändert. Wird mit einer früheren Version auf den Quellcode zugegriffen, werden die Codepage-Informationen einfach ignoriert.

Stichwortverzeichnis

B

bidirectional language support, 61

C

CALLNAT
    statement, 49

D

DEFINE PRINTER
    statement, 49

E

EXAMINE
    statement, 47

L

logical condition criteria
    U format, 50

M

MOVE ENCODED
    statement, 46
MOVE NORMALIZED
    statement, 46

P

PARSE JSON
    statement, 48
PARSE XML
    statement, 49
print files
    Unicode support, 69
profile parameters
    important for Unicode, 26

R

REQUEST DOCUMENT
    statement, 49

S

SAGICUA9, 12
session parameters
    important for Unicode, 51
statements
    important for Unicode, 45
system variables
    important for Unicode, 50

W

work files
    Unicode support, 69