Natural for Windows

Unicode およびコードページのサポート

バージョン 6.3.3

October 2008

Natural

This document applies to Natural バージョン 6.3.3 and to all subsequent releases.

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

目次

1 Unicode およびコードページのサポート

このドキュメントでは、Windows、UNIX、および OpenVMS プラットフォームでの、Natural による Unicode およびコードページのサポートについて説明します。また、Windows、UNIX、および OpenVMS プラットフォームでの、Natural による双方向言語およびダブルバイト文字セットのサポートについても説明します。

このドキュメントは次の項目で構成されています。

はじめに	コードページおよび Unicode 標準に関する全般的な情報、および Natural での Unicode およびコードページのサポートに関する全般的な情報
Natural プログラミング言語での **Unicode** およびコードページのサポート	U フォーマットに関する情報、および Unicode およびコードページのサポートを提供するステートメント、論理条件基準、システム変数、ラージ変数およびダイナミック変数、およびセッションパラメータに関する情報
Unicode／コードページ環境の設定と管理	ICU ライブラリに関する情報、Unicode およびコードページのサポートを提供するプロファイルパラメータに関する情報、およびコードページデータのエンコードに関する情報
開発環境	環境をカスタマイズする方法および Natural エディタによる Unicode の処理
Natural アプリケーションの入力／出力処理	Unicode データの表示および入力方法 SPoD およびランタイム環境で使用される Web I/O インターフェイスクライアントに関する情報
Unicode データストレージ	データベースアクセスに関する情報、および Unicode およびコードページのサポートを提供するワークファイルタイプおよび出力ファイルに関する情報
プラットフォームの相違	Windows、UNIX、および OpenVMS プラットフォームでの処理の相違。
既存アプリケーションの移行	既存アプリケーションへの Unicode の影響。 既存オブジェクトの移行、既存アプリケーションへの Unicode サポートの追加、および Natural リモートプロシージャコール（RPC）の移行方法。

	特別な考慮事項と制限事項	Windows、UNIX、および OpenVMS プラットフォームでの重要な情報および制限
	双方向言語サポート	Windows、UNIX、およびOpenVMSプラットフォームでのNaturalによる双方向言語のサポート方法
	ダブルバイト文字サポート	Natural によるダブルバイト文字セットのサポート方法
	よくある質問	よくある質問への回答

2 Introduction

このchapterでは、次のトピックについて説明します。

- コードページおよび Unicode について
- Natural での Unicode およびコードページのサポートについて

コードページおよび Unicode について

従来のコードページは、特定の言語または共通スクリプトを共有する言語グループをサポートする、特定の順序で並べられた、選択された文字コードのリストです。 コードページには、最大で256個の文字コードを含めることができます。中国語や日本語など、256文字よりも多い文字セットの場合は、ダブルバイトコード単位処理（DBCS）が使用されます。DBCS コードページは実際にはマルチバイトエンコードであり、1バイトおよび2バイトコードポイントが混在します。

コードページには、同じデータ文字列内に異なる言語を保存できないという固有の短所があります。 Unicode は、この制限をなくすために設計されました。データへのアクセスに使用されるプラットフォーム、プログラム、または言語に依存しない標準エンコードが、すべての文字セットに提供されます。 Unicode では、すべての文字に一意の番号が付けられます。

Unicode 標準によって定義された各コード要素に 1 つの番号が割り当てられます。 これらの番号それぞれは「コードポイント」と呼ばれ、テキスト内で参照されるときは、"U" の後に 16 進形式でリストされます。 例えば、コードポイント "U+0041" は、16 進数 "0041"（10 進数 "65" と同じ）です。 これは、Unicode 標準の文字 "A" を表し、「LATIN CAPITAL LETTER A」という名前です。

Unicode標準では、同じデータをバイト、ワード、またはダブルワード指向フォーマットで転送することを可能にする 3 つのエンコード形式が定義されています。 「コード単位」は、特定のエンコードで文字を表すことができる最小のビット組み合わせです。 Unicode 標準では、UTF-8 エンコード形式で 8 ビットのコード単位、UTF-16 エンコード形式で 16 ビットのコード単位、UTF-32 エンコード形式で 32 ビットのコード単位が使用されます。 3つのエンコード形式すべてで*同じ*共通文字レパートリがエンコードされるため、データを欠落させずに相互に効率的に変換できます。

Natural のコンテキストでは、これらのエンコード形式のうちの UTF-16 および UTF-8 の 2 つが関係します。 Natural では、ランタイム時の Unicode 文字列のコーディングに UTF-16 が使用され、ファイル内の Unicode データのコーディングに UTF-8 が使用されます。 UTF-16 は、エンディアンに依存する2バイトエンコードです。使用されるエンディアンフォーマットは、プラットフォームによって異なります。 UTF-8 は 1 バイトエンコードです。

Unicode の詳細については、Unicode Consortium の Web サイト ***http://www.unicode.org/*** を参照してください。

Note: Unicode コードポイントに関する情報を取得するには、Natural for Windows で使用可能な SYSCP ユーティリティを使用します。

Natural での Unicode およびコードページのサポートについて

以前のバージョンのNaturalでは、コードページの文字のみがサポートされていました。Natural バージョン 4.2（メインフレーム）、6.2（Windows および UNIX）、および 6.3（OpenVMS）から、Unicode がサポートされています。

Unicodeのサポートに関して、Naturalデータフォーマット U が導入されており、固有のステートメント、パラメータ、システム変数などがあります。詳細については、このドキュメントの該当する箇所を参照してください。

現在、既存のデータのほとんどをコードページフォーマットで使用できます。 このデータをUnicodeに変換する場合は、正しいコードページを使用する必要があります。 Naturalでは、次の複数のレベルで正しいコードページを定義できます。

- Naturalでデフォルトコードページが定義されていない場合は、システムコードページが使用されます。
- Natural パラメータ `CP` が定義されている場合は、デフォルトコードページが使用されます。システムコードページは上書きされます。
- ソースなどに対して定義されているオブジェクトコードページによって、このオブジェクトのデフォルトコードページは上書きされます。

1 つのアプリケーションで Unicode 文字列とコードページ文字列を使用する場合は、Natural によって、必要に応じて（例えば、データを移動または比較する場合に）暗黙的に変換されます。明示的な変換は、ステートメント `MOVE ENCODED` を使用して実行できます。

ほとんどの場合、Unicodeのサポートを必要としない既存アプリケーションは、変更なく実行されます。 Windows、UNIX、および OpenVMS プラットフォームでは、既存のソースが異なるコードページでエンコードされる場合に、変更が必要な場合があります。 詳細については、このドキュメントの「*既存アプリケーションの移行*」を参照してください。

既存アプリケーションを実行し、そのアプリケーションを変更しないでUnicodeデータもサポートすることはできません。 Natural データフォーマット U をアプリケーションに導入する必要があります。ほとんどの場合、A フォーマット定義を U フォーマット定義で置き換えるだけでは不十分です。 文字列の特定のメモリレイアウトを前提とするすべてのコード（例えば、英数字から数字フォーマットへの `REDEFINE`）を変更する必要があります。

Unicode文字は、変数名、オブジェクト名、およびライブラリ名で使用することはできません。

Unicode ベースのデータは、Adabas でサポートされています。

Natural では、Unicode の照合および変換について、International Components for Unicode（ICU）ライブラリが使用されます。 詳細については、***http://icu.sourceforge.net/userguide/*** を参照してください。 このドキュメントの「***ICU ライブラリ***」も参照してください。

3 Natural プログラミング言語での Unicode およびコードページのサポート

このchapterでは、次のトピックについて説明します。

Unicode ベースのデータ用の Natural データフォーマット U

Natural では、フォーマット U および U 定数を使用して Unicode 文字列を指定できます。

- フォーマット **U**
  フォーマットUを使用して、Unicode文字列を保持するデータを定義できます。 Naturalデータフォーマット U は、内部的には UTF-16 です。

  『*プログラミングガイド*』の「*ユーザー定義変数のフォーマットおよび長さ*」も参照してください。
- **U** 定数
  接頭辞 "U" を使用して、Unicode 定数を定義できます。 次に例を示します。

```
U'Äpfel'
```

  接頭辞 "UH" は、Unicode 定数を 16 進形式で定義するために使用できます。 Unicode 標準で定義されているように、4桁の16進数は1つのUTF-16コード単位を表します。 したがって、全体の長さは 4 の倍数になります。 例えば、次の 16 進形式が必要であるとします。

```
U'Äpfel'
```

  "Ä"、"p"、"f"、"e"、および "l" の UTF-16 コード単位
  （"U+00C4"、"U+0070"、"U+0066"、"U+0065"、および "U+006C"）が必要であり、これらを次の 16 進数の文字列に結合する必要があります。

```
UH'00C40070006600650006C'
```

  『*プログラミングガイド*』の「*Unicode 定数*」も参照してください。

データフォーマット U は、エンディアンに依存します。 フォーマット B と U 間で移動する場合は、これを考慮する必要があります。

U と **A** の比較

A フォーマットと比較した U フォーマットの利点は、さまざまな言語の文字の任意の組み合わせを保持でき、インストールされているシステムコードページに依存しない点です。 さらに、Uフォーマットでは、異なるプラットフォーム間でのデータの共有が簡単です。変換（例えば、EBCDIC から ASCII へ）は必要ありません。

一方、U フォーマットデータは、A フォーマットデータよりも多くのメモリが消費される場合があります。 ほとんどの文字列をシングルバイトエンコードで表すことができる言語の場合、

Uフォーマットでは、以前に必要だったスペースの2倍のスペースが文字列によって占められるようになります。 ただし、東アジア言語の場合は、通常、メモリ消費は増加しません。

ステートメント

基本的に、A フォーマットを使用できるステートメントのほとんどで、U フォーマットを使用できます。 ただし、ステートメントのオペランドとして Natural オブジェクト名が与えられる場合（CALLNAT ステートメントなど）、Natural オブジェクト名は A フォーマットであるため、U を使用できません。 該当するステートメントの詳細については、『ステートメント』ドキュメントを参照してください。

基本的に、A および U フォーマットは、1 つのステートメント内で一緒に使用できます。ただし、1つのステートメント内では、いずれか1つのフォーマットのみを使用することをお勧めします。 両方のフォーマットが一緒に使用されると、すべての変数を統一フォーマットに変換する必要があり、変換エラーが発生する場合があります。

Unicode では、次のステートメントが重要です。

- MOVE NORMALIZED
- MOVE ENCODED
- EXAMINE
- PARSE XML
- REQUEST DOCUMENT
- DEFINE PRINTER
- CALLNAT (RPC)

MOVE NORMALIZED

Unicode での正規化：バイナリでの等価比較が可能な形式へデータを変換するために、同等のシーケンスの代替表現をテキストデータから削除するプロセス。 Unicode 標準では、さまざまな正規化形式が定義されています。 Unicode 文字列の正規分解の結果の正規化形式の後に、可能であればプライマリ合成による分解されたすべてのシーケンスに置き換わるものを付けた形式は、「合成済み正規化形式」（NFC）と呼ばれます。

Natural では、Unicode 文字を部分的に切り捨てることなく文字列処理を実行できるように、すべての Unicode データは NFC フォーマットであることが前提となっています。 Natural 変換処理では、結果として生じる Unicode 文字列は NFC であることが保証されます。 Natural の外部から Unicode データを受け取り、そのデータが NFC フォーマットであることが保証されない場合は、MOVE NORMALIZED ステートメントを適用できます。

例：

文字シーケンス	NFC
ê (U+00EA)	ê (U+00EA)
e (U+0065) + ^ (U+0302)	ê (U+00EA)

Note: NFC フォーマットの 2 つ以上の文字列を連結した場合、NFC フォーマットにならない場合があります。

MOVE ENCODED

MOVE ステートメントを使用して文字列をUからAへ、またはその逆に移動するときに、Unicodeとデフォルトコードページ間の暗黙的な変換が実行されます。

さらに、異なるコードページ間の変換、または任意のコードページから Unicode への変換、およびその逆への変換に、MOVE ENCODED ステートメントを使用できます。 これは、Natural の外部からのデータがデフォルトコードページとは異なるコードページでコード化されている場合に役立ちます。 しかし、デフォルトコードページと Unicode 間の変換についても、GIVING 節での潜在的な変換エラーを取得する場合は、このステートメントを使用できます。この場合は、CPCVERR が "ON" に設定されている場合に、MOVE ステートメントはランタイムエラーで停止します。

文字を変換できない場合に、この文字に対して置換文字が使用されるか変換が失敗するかは、CPCVERR パラメータの設定によって異なります。 Windows、UNIX、および OpenVMS プラットフォームでは、Unicode からデフォルトコードページ（CP）への変換について ICU によって定義されたデフォルトの置換文字は、プロファイルパラメータ SUBCHAR を使用して変更できます。

このステートメントは、U データから UTF-8 フォーマットへの変換についても使用できます。

Note: デフォルトコードページとは異なるコードページにデータを変換する場合は、このデータを I/O で使用しないことをお勧めします。 I/O は、デフォルトコードページでのみ意味があります。

EXAMINE

「grapheme」は、ユーザーが一般に文字と見なすものです。ほとんどの場合、Unicode コードポイントは書記素ですが、書記素が複数の Unicode コードポイントで構成される場合もあります。 例えば、1 つの基底文字と 1 つ以上の結合文字のシーケンスは書記素です。

例："a" (U+0061) + "." (U+0323) + "^" (U+0302) は、次のように表示される 1 つの書記素を定義します。

ậ

Note: 基底文字／結合文字のシーケンスが正規化される場合、このことはシーケンスが常に合成済み文字によって置き換えられることを意味しません。すべての文字に合成済みフォーマットがあるわけではないためです。

「補助コードポイント」 は、"U+10000" と "U+10FFFF" の間の Unicode コードポイントです。補助コードポイントは UTF-16 であり、2 つのコード単位で構成されるサロゲートペアによって表されます。ペアの最初の値は「上位サロゲートコード単位」、2番目は「下位サロゲートコード単位」 です。 このような文字は一般的にはまれですが、中国語や日本語の人名の一部などで使用される場合があります。したがって、東アジアの国の行政アプリケーションでは、一般にこれらの文字のサポートが必要です。

EXAMINE とその SUBSTRING オプションなどの文字列操作ステートメントは、UTF-16 コード単位を処理します。コードによって書記素またはサロゲートペアが分離されないようにすることは、ユーザーの責任です。

ただし、EXAMINE ステートメントの節 CHARPOSITION および CHARLENGTH（「*構文 3 - Unicode 書記素用の EXAMINE*」を参照）を使用して、書記素の UTF-16 コード単位での開始位置および長さを求めることができます。 結果の値は、SUBSTRING 呼び出しに使用できます。 これらの節を使用して、書記素ごとに文字列をスキャンできます。

例：

```
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'aอิcอิbหัcยี่d' 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
```

上記のプログラム例の出力は、次のとおりです。

```
Page     1                                                  05-12-15  09:33:49

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

aปีcอี่bนัcปี่d               1                1              1
aปีcอี่bนัcปี่d               2                2              2
aปีcอี่bนัcปี่d               3                4              1
aปีcอี่bนัcปี่d               4                5              3
aปีcอี่bนัcปี่d               5                8              1
aปีcอี่bนัcปี่d               6                9              3
aปีcอี่bนัcปี่d               7               12              1
aปีcอี่bนัcปี่d               8               13              3
aปีcอี่bนัcปี่d               9                0              0
```

PARSE XML

XML 文書では、文書のエンコードに関する情報を XML 文書ヘッダーに含めることができます（例えば、<?xml version="1.0" encoding="UTF-8" ?>）。XML 文書にこの情報が含まれている場合、Windows、UNIX、および OpenVMS プラットフォームでの XML 文書の解析には、受け取りフィールドがフォーマット U ではない場合は、XML 文書ヘッダー内で指定されたコードページから Natural の現在のコードページへの変換が常に含まれます。

詳細については、PARSE XML ステートメントの説明を参照してください。

REQUEST DOCUMENT

REQUEST DOCUMENT ステートメントを使用したデータ転送では、通常、コードページ変換は呼び出されません。送信データ／受信データを特定のコードページでエンコードする場合は、REQUEST DOCUMENT ステートメントの DATA ALL 節／RETURN PAGE 節を使用して指定できます。

詳細については、REQUEST DOCUMENT ステートメントの説明を参照してください。

DEFINE PRINTER

メインフレームプラットフォームでは、DEFINE PRINTER ステートメントに CODEPAGE 節があり、出力レポートデータを現在のランタイム設定とは異なるコードページに変換できます。Windows、UNIX、および OpenVMS プラットフォームでは、DEFINE PRINTER ステートメントにそのような節はありません。CODEPAGE 節が定義された場合、Windows、UNIX、および OpenVMS プラットフォームでは無視されます。

CALLNAT（**RPC**）

RPC 経由での Unicode フォーマットでのデータ交換がサポートされています。 CALLNAT ステートメントの説明を参照してください。

U データがビッグエンディアンエンコードのプラットフォームからリトルエンディアンエンコードのプラットフォームへ、またはその逆に送信される場合、エンコードは受信プラットフォームのエンコードに準拠するように変更されます。例えば、リトルエンディアンエンコードの U データがビッグエンディアンのプラットフォームに到着した場合、このデータはビッグエンディアンエンコードに変換されてからプログラムに渡されます。 このデータが送り返されるときは、リトルエンディアンエンコードに戻されます。

論理条件基準

論理条件基準では、Unicode のオペランドを英数字およびバイナリのオペランドとともに使用できます。 すべてのオペランドが Unicode のオペランド（フォーマット U）というわけではない場合は、2 番目以降のオペランドはすべて最初のオペランドのフォーマットに変換されます。バイナリのオペランド（フォーマット B）が 2 番目以降のオペランドとして指定された場合、バイナリのオペランドの長さは偶数である必要があります。バイナリのオペランドには Unicode コードポイントが含まれていると想定されます。

最初のオペランドが Unicode のオペランド（フォーマット U）であるために、比較が Unicode 比較として実行される場合、ICU 照合アルゴリズムが使用されます。 ICU アルゴリズムでは、単純なバイナリ比較は実行されません。 そのため、例えば次のような結果になります。

- "U+0000" などの一部のコードポイントは、比較プロセスで無視されます。
- 組み合わされた文字は、等価の 1 つのコードポイントと等しいとみなされます。例えば、"U+00E4" によって表されるドイツ語の文字 "ä" と、コードポイント "U+0061" および "U+0308" の組み合わせは、ICU によって等しいとみなされます。

Note: 英数字と Unicode のオペランドの比較は、フィールドのシーケンスに応じて、異なる結果となる場合があります。

『*プログラミングガイド*』の「*論理条件基準*」も参照してください。

システム変数

このsectionでは、次のトピックについて説明します。

- *CODEPAGE
- *LOCALE

*CODEPAGE

システム変数 *CODEPAGE は、Unicode とコードページフォーマット間の変換に現在使用されているコードページの IANA 名を返すために使用されます。

*LOCALE

システム変数 *LOCALE には、現在のロケールの言語および国が含まれています。

ラージ変数およびダイナミック変数

U フォーマットは、ラージ変数およびダイナミック変数に使用できます。 ダイナミック U 変数の場合、*LENGTH によって UTF-16 コード単位の数が返されます。

『*プログラミングガイド*』の「*ダイナミック変数およびフィールドについて*」も参照してください。

セッションパラメータ

次のセッションパラメータを使用できます。

パラメータ	説明
DL	フォーマット A または U のフィールドの表示長を指定します。『*プログラミングガイド*』の「*出力の表示長 - DL パラメータ*」も参照してください。
EMU *	Unicode での編集マスク
ICU *	Unicode での挿入文字
LCU *	Unicode での先頭文字
TCU *	Unicode での末尾文字

上記の表でアスタリスク（*）マークが付いたセッションパラメータは、Windows、UNIX、および OpenVMS プラットフォームでのみ使用できます。

DL と AL の比較

Naturalが Unicode 対応ではなかった間は、英数字フィールドの長さは、フィールドを表示するために必要な列の数（表示列の数）と常に同じでした。 このことは、DBCS コードページを使用する東アジア言語についても該当しました。A フォーマットフィールドは、半分の文字のみを保持できます。例えば、A10 は A5 になります。

例：

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

上のコードの出力結果は、次のとおりです。

```
Page      1 ...

computer
電腦系統
```

U フォーマットフィールドでは、フィールドの長さと表示列の数は同じではなくなります。 U の文字は、狭い幅（ラテン文字など）か、広い幅（中国語文字など）か、または幅を持たない（結合文字など）場合があります。 したがって、U フィールドが必要とする表示列の数はまったくわかりません。これは、フィールドの内容によって異なります。 Naturalでは、画面上に予約する必要がある列の数を自動的に決定できません。最大サイズを想定するとラテン出力で大きな差異が発生し、最小サイズを想定すると中国語出力が完全には表示されない場合があります。したがって、Natural のプログラマは、フィールドの表示幅を定義する必要があります。これは、DL パラメータを使用して行います。 AL パラメータは、この目的には使用できません。このパラメータでは、フィールドの定義された長さを超える部分が切り取られるためです。Uフィールドから文字を切り取るのではなく、次のフィールドの開始位置のみを定義する必要があります。

例：

```
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
```

上記のコードの結果は、前と同じ出力になります。

```
Page      1 ...

computer
電腦系統
```

Windows では、出力ウィンドウを使用してローカルで、または Web I/O インターフェイスクライアントを使用してリモート開発環境で、DL パラメータに定義された値がフィールドの実際の表示幅よりも小さいフィールド内をスクロールできます。

EMU、ICU、LCU、TCU と EM、IC、LC、TC の比較

パラメータ EMU、ICU、LCU、および TCU（Windows、UNIX、および OpenVMS プラットフォームでのみ使用可能）を使用すると、デフォルトコードページに含まれていない文字を使用できます。 文字は、生成されたプログラムに Unicode フォーマットで保存されます。 これらのパラメータは、すべてのフィールドフォーマットで使用できます。

パラメータ EM、IC、LC、および TC も、U フォーマットフィールドで使用できます。 デフォルトコードページに含まれている文字が他のコードページではエンコードが異なる場合に、これらのパラメータも便利な場合があります。 例えば、ユーロ記号（€）は、"windows-1252"（Latin 1）コードページではコードポイント "0x80" ですが、"windows-1251"（キリル語）コードページではコードポイント "0x88" です。 したがって、Unicode パラメータを使用することによって、PC にインストールされているコードページにかかわらず、ユーロ記号は常に正しく表示されるようになります。

EMU の例：

```
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
```

上のコードの出力結果は、次のとおりです。

```
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€
```

サンプルプログラム

ライブラリ SYSEXPG に、Natural での Unicode およびコードページのサポートのサンプルプログラムがあります。

- UNICOX01 は、すべての Unicode 文字をリストします。
- UNICOX02 は、Unicode 文字をコードポイントに、またはその逆に変換します。
- CODEPX01 は、すべてのコードページ、そのコードページが Natural でサポートされているかどうか、およびそのコードページが使用するエンコードをリストします。 サポートされるすべてのコードページについて、そのコードページの文字をリストするサービスおよびコードページの文字列を 16 進表示に変換したり、その逆に変換したりするサービスを提供します。
- CODEPXL1 は、任意の 1 バイトコードページのすべての文字をリストします。

- CODEPXL2 は、任意の 2 バイトコードページのすべての文字をリストします。
- CODEPXC1 は、任意のコードページの文字列を 16 進表示に、およびその逆に変換します。

4 Unicode／コードページ環境の設定と管理

このchapterでは、次のトピックについて説明します。

ICU ライブラリ

Windows、UNIX、および OpenVMS プラットフォーム

ICU ライブラリは、ICU 変換および照合データのフルセットとともに常にインストールされます。 コンフィグレーションファイル *NATCONV.INI* の設定値が A フォーマットに適用されます。Uフォーマットに対して、対応するチェック（文字が大文字に変換されるときなど）がICUライブラリによって行われます。

Note: ICU バージョンおよびサポートされるコードページに関する情報を取得するには、Natural for Windows で使用可能な `SYSCP` ユーティリティを使用します。

プロファイルパラメータ

このセクションでは、Unicodeおよびコードページのサポートとともに使用されるプロファイルパラメータについて説明します。 すべてのプラットフォームですべてのプロファイルパラメータが使用できるわけではありません。

すべてのプラットフォーム

次のプロファイルパラメータは、すべてのプラットフォームで使用できます。

パラメータ	説明
`CP`	Natural のデフォルトコードページを定義します。 このコードページは、単一のオブジェクト（Natural ソースなど）に対して定義されたコードページがない場合に、ランタイム環境および開発環境に対して使用されます。 プラットフォームに適したコードページのみを使用できます。 つまり、例えば、EBCDIC コードページは、Windows、UNIX、または OpenVMS プラットフォームには定義できません。。 **注意:** Natural for Windows/UNIX バージョン 6.2、Natural for OpenVMS バージョン 6.3、および Natural for Mainframes バージョン 4.2 以降、Natural RPC で使用される既存の `CP` パラメータは `CPRPC` に名前が変更されています。

パラメータ	説明
CPCVERR	Unicode からコードページへ、コードページから Unicode へ、または 1 つのコードページから別のコードページへ変換するときに発生する変換エラーが、Natural エラーになるかどうかを指定します。 このパラメータは、Natural ソースをソースエリアにロードするとき、または Natural ソースをカタログするときの Natural ソースの変換には関連しません。
CPOBJIN	データのバッチ入力ファイルのエンコードに使用するコードページを指定します。このファイルは、Natural プロファイルパラメータ CMOBJIN（Windows、UNIX、および OpenVMS）で定義されています。
CPPRINT	バッチ出力ファイルのエンコードに使用するコードページを指定します。このファイルは、Natural プロファイルパラメータ CMPRINT（Windows、UNIX、および OpenVMS）で定義されています。
CPSYNIN	コマンドのバッチ入力ファイルのエンコードに使用するコードページを指定します。このファイルは、Natural プロファイルパラメータ CMSYNIN（Windows、UNIX、および OpenVMS）で定義されています。
SRETAIN	すべての既存のソースを元のエンコードフォーマットで保存する必要があることを指定します。「*環境のカスタマイズ*」も参照してください。

以下の項目も参照してください。

- Natural for Windows、Natural for UNIX、および Natural for OpenVMS については、『*オペレーション*』ドキュメントの「*バッチモードでの Natural*」セクションにある「*入力および出力ファイル用のコードページ*」
- 有効なコードページについては、***http://www.iana.org/assignments/character-sets***

Windows、UNIX、および OpenVMS プラットフォーム

次のプロファイルパラメータは、Windows、UNIX、および OpenVMS プラットフォームでのみ使用できます。

パラメータ	説明
SUTF8	Natural ソースが保存されるときに使用されるデフォルトのフォーマットを指定します。 **注意:** UNIX および OpenVMS では、このパラメータは SPoD 環境でのみ使用できます。
SUBCHAR	Unicode からデフォルトコードページへの変換用の置換文字を指定します。SUBCHAR が "OFF" の場合は、ICU によって定義されたデフォルトの置換文字が使用されます。 **注意:** SUBCHAR は、コードページから Unicode への変換、または Unicode から現在のコードページとは異なるコードページへの変換には影響しません。
WEBIO	Web I/O インターフェイスクライアント（Unicode をサポート）または端末エミュレーションウィンドウ（非 Unicode 対応）が入出力に使用されるかどうかを指定します。 ローカルな Windows 環境では、出力ウィンドウ（Unicode 対応）が使用されます。

パラメータ	説明
	リモート Windows 環境では、このパラメータの設定に関係なく、Web I/O インターフェイスクライアントが常に使用されます。

エンコード情報

コードページデータのエンコードは、次に示す異なるレベルで指定できます。

- レベル 1 - デフォルトコードページ
- レベル 2 - 単一のオブジェクトのコードページ

レベル 1 - デフォルトコードページ

デフォルトコードページは、CP パラメータを使用して定義できます。 Windows、UNIX、および OpenVMS プラットフォームでは、システムコードページを上書きして、すべてのコードページデータに対して有効になります。

レベル 2 - 単一のオブジェクトのコードページ

Natural ソース、バッチ入力（CPOBJIN、CPSYNIN）、および出力ファイル（CPPRINT）に対してコードページを定義できます。

また、Windows、UNIX、および OpenVMS プラットフォームでは、タイプ ASCII、圧縮 ASCII、Unformatted、および CSV のワークファイルに対してコードページを定義できます。『*コンフィグレーションユーティリティ*』ドキュメントの「*ワークファイル割り当て*」を参照してください。

コードページがオブジェクトレベルで定義された場合、このコードページによってデフォルトコードページは上書きされます。

Note: Windows、UNIX、および OpenVMS プラットフォームでは、すべてのオブジェクトに対して正しいコードページが定義されることが重要です。 詳細については、「*既存アプリケーションの移行*」を参照してください。

エンコード情報を持つ Natural オブジェクトの展開

Windows、UNIX、および OpenVMS プラットフォーム

エンコード情報がすでに定義された Natural オブジェクトを展開する場合、エンコード情報はファイル *FILEDIR.SAG* に保存されており、Natural の外部からオブジェクトファイルのみを展開した場合はエンコード情報が失われることに注意する必要があります。

Natural オブジェクトを展開する場合、次のようにエンコード情報を保持できます。

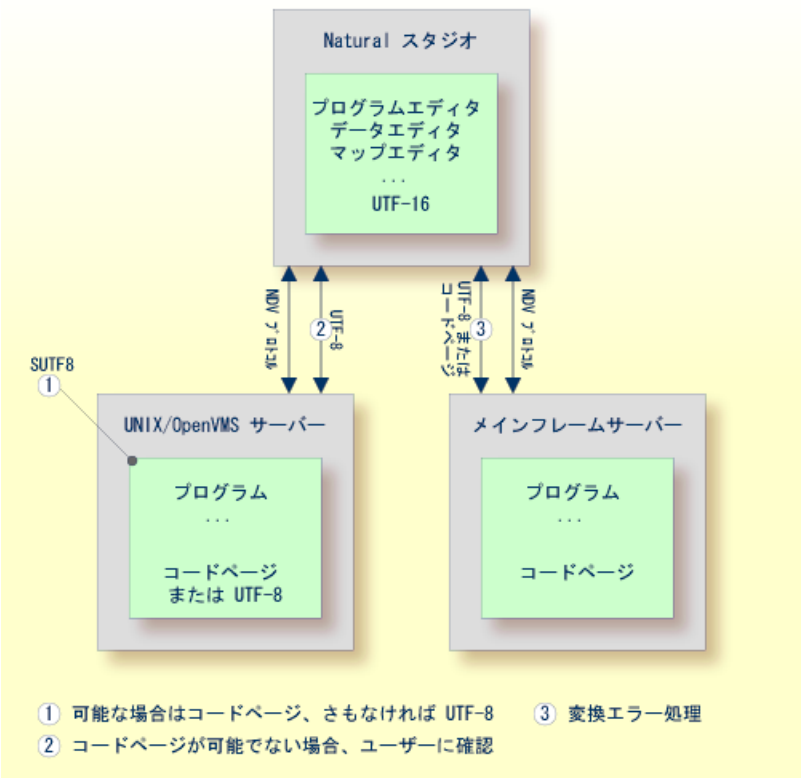
- ライブラリ全体をコピーできます。 その後、ライブラリのコピーをすべての Windows、UNIX、およびOpenVMSプラットフォームに分散できます。この場合は、元のコードページが維持されます。ライブラリをWindowsからUNIXまたはOpenVMSにコピーする場合は、Natural for UNIX または Natural for OpenVMS のネイティブエディタでオブジェクトを開くことができない場合があることに注意する必要があります。これらのエディタでは、デフォルトコードページを持つオブジェクトのみを開くことができるためです。
- エンコード情報を保持するオブジェクトハンドラを使用できます。 この場合は、元のコードページが維持されます。 Windows ライブラリが UNIX または OpenVMS でアンロードされる場合は、Natural for UNIX または Natural for OpenVMS のネイティブエディタでオブジェクトを開くことができない場合があることに注意する必要があります。これらのエディタでは、デフォルトコードページを持つオブジェクトのみを開くことができるためです。
- Natural スタジオを使用して、オブジェクトをコピーして貼り付けることができます。 SPoD 環境で、ターゲット環境がソース環境とは異なるプラットフォーム上にある場合、Naturalはターゲット環境のデフォルトコードページを使用してオブジェクトを保存しようとします。これが不可能な場合、オブジェクトは UTF-8 フォーマットで保存されます。 UNIX および OpenVMS ターゲットの場合、このことによって、ソースのすべての文字が UNIX または OpenVMSサーバーのデフォルトコードページで使用可能な場合は、Natural for UNIX または Natural for OpenVMS のネイティブエディタでオブジェクトを開けます。

5 開発環境

このchapterでは、次のトピックについて説明します。

開発環境

Unicode アプリケーションの開発環境は、Natural Single Point of Development（SPoD）です。



SPoD 環境では、Natural 開発サーバー（NDV）上の Unicode アプリケーションの Natural オブジェクトは、Natural スタジオを使用して変更できます。サーバーによってサポートされている場合、ソースはクライアントとサーバー間で UTF-8 フォーマットで交換されます。

UNIX および OpenVMS 用の NDV サーバーでは、プロファイルパラメータ SUTF8 の設定によって、Natural オブジェクトをサーバーに保存するときに使用されるフォーマットが決まります。これは、ローカルの Windows の場合と同様に処理されます。

メインフレーム用の NDV サーバーでは、オブジェクトはデフォルトまたは元のエンコードで保存されます。

環境のカスタマイズ

Windows、UNIX、およびOpenVMSプラットフォームでは、Naturalコードを変更する前に、環境に対して正しいデフォルトコードページを定義することが重要です。詳細については、「*既存アプリケーションの移行*」を参照してください。

異なる言語の文字をソースに保存する場合、Windows、UNIX、またはOpenVMSプラットフォームでUTF-8フォーマットでソースを保存するか、ソースで16進UH定数を使用する必要があります。 プロファイルパラメータ `SUTF8` および `SRETAIN` を使用して、どのフォーマットでソースを保存するかを制御できます。 次の表に、いくつかの状況と推奨設定を示します。

Note: UNIXおよびOpenVMSでは、パラメータ `SUTF8` はSPoD環境でのみ使用できます。

状況	設定	効果
ソースはWindows上にあります。U定数が必要です。	`SUTF8=ON, SRETAIN=OFF`	Natural 6.2以降で保存する場合、すべてのソースはUTF-8フォーマットで保存されます。 新しいソースはUTF-8フォーマットで作成されます。 すべての文字を1つのソースに保存できます。
ソースはWindows、UNIX、またはOpenVMS上にあります。U定数が必要であり、SPoDが開発に使用されます。	`SUTF8=ON, SRETAIN=ON`	元のコードページへの変換が不可能になった場合、すべてのソースはUTF-8フォーマットで保存されます。可能な場合には、ソースのコードページは変更されません。新しいソースはUTF-8フォーマットで作成されます。 すべての文字を1つのソースに保存できます。UTF-8フォーマットのソースは、SPoDでのみ変更できます。Natural for UNIXまたはNatural for OpenVMSエディタでは処理できなくなります。
ソースはWindows、UNIX、またはOpenVMS上にあります。U定数は必要ありません。	`SUTF8=OFF, SRETAIN=ON`	すべてのソースは元のコードページで保存されます。 新しいソースは、サーバーのデフォルトコードページで保存されます。1つのソースに保存できるのは、ソースコードページからの文字のみです。 ソースは、引き続きNatural for UNIXまたはNatural for OpenVMSエディタで処理できます。
ソースはWindows、UNIX、OpenVMS、またはメインフレーム上にあります。U定数が必要であり、SPoDが開発に使用されます。	`SUTF8=OFF, SRETAIN=ON`	すべてのソースは元のコードページで保存されます。 新しいソースは、サーバーのデフォルトコードページで保存されます。1つのソースに保存できるのは、ソースコードページからの文字のみです。 ソースは、引き続きNatural for UNIX、Natural for OpenVMS、およびNatural for Mainframesエディタで処理できます。 すべてのUnicode定数は、16進定数（UH）として定義される必要があります。

パラメータ `SUTF8` が "OFF" に設定されており、異なる文字セットの文字を含むが、まだUTF-8フォーマットで保存されていないソースを格納する場合、生成プログラムは作成されるが、ソースは保存不可能であり、そのためにソースは変更されないままである場合があります。これは、

異なる文字セットの文字がコメントまたは U 定数で使用される場合に発生します。 このため、異なる文字セットの文字を含むソースを作成する場合、およびソースをメインフレームプラットフォームに分散する必要がない場合は、パラメータ SUTF8 を "ON" に設定することをお勧めします。

パラメータ SRETAIN が "OFF" に設定されている場合、すべてのソースはデフォルトコードページで保存されます。 この設定には注意が必要です。以前の Natural バージョンで作成されたソースがある場合、この設定によってコードページ情報が不適切になる可能性があるためです。 この場合、ソースのエンコード情報は割り当てられておらず、ソースは常に現在のコードページで開かれます。 現在のコードページがソースの正しいエンコードではない場合でも、機能する場合があります。 この場合、言語固有の一部の文字は正しく表示されません。 そのようなソースが間違ったコードページで開かれ、SRETAIN が "ON" に設定された状態で保存された場合、ソースに対してエンコードは保存されません。ソースは、Natural が正しいデフォルトコードページで開始された場合に、後で正しく開くことができます。 ただし、SRETAIN が "OFF" に設定された状態でソースを保存すると、現在のコードページがソースのエンコードとして保存されます。その後、ソースはこのコードページでのみ開かれます。このため、この設定は、すべての Natural ソースが現在のコードページでエンコードされていることが確かな場合にのみ使用する必要があります。

『*コンフィグレーションユーティリティ*』ドキュメントの「*地域の設定*」も参照してください。

エディタ

Natural for Windows エディタは、Unicode に完全に対応しています。 SPoD 経由で、メインフレーム、UNIX、および OpenVMS ソースに対しても使用できます。 Natural for Mainframes、Natural for UNIX、および Natural for OpenVMS で提供されるエディタは、Unicode 対応にはならない予定です。

Natural スタジオ（Natural for Windows）のエディタを使用してソースが開かれるとき、ソースの内容は、対応するコードページから Unicode に変換されてから、エディタにロードされます。 このことによって、システムコードページに含まれない文字がソースに含まれている場合でも、すべての文字を正しく表示できることが保証されます。ソースのコードページから Unicode への変換が失敗した場合は、エラーが表示され、エディタは開かれません。この場合、ユーザーはソースの正しいエンコードを定義する必要があります。 ソースのエンコードは、［プロパティ］ダイアログボックスで変更できます（『*Natural スタジオの使用*』ドキュメントの「*ノードのプロパティ*」を参照）。

Windows、UNIX、および OpenVMS ソースの場合、Natural for Windows エディタによって、異なる言語の文字を含むソースを UTF-8 フォーマットで保存できます。 メインフレームでは、UTF-8 ソースを保存できません。

Note: UNIX または OpenVMS ソースを UTF-8 フォーマットで、またはデフォルトコードページとは異なるコードページを使用して保存した場合、ソースは Natural for UNIX または Natural for OpenVMS のネイティブエディタで開くことができなくなります。 メイ

ンフレームソースは、異なるコードページを使用して保存し、Natural for Mainframes のネイティブエディタで編集できます。

プログラムおよびソース内で Unicode 文字列を使用しない場合でも、Unicode 対応エディタには、インストールされているシステムコードページに関係なく、すべてのコードページのソースを記述できるという利点があります。 例えば、"windows-1252"（Latin 1）コードページをインストールしている場合に、キリル語文字を含むプログラムを記述し、このプログラムを"windows-1251"（キリル語）コードページで保存できます。 ［名前をつけて保存］ダイアログボックスで、コードページ "windows-1251" を選択するだけです（『*Natural スタジオの使用*』ドキュメントの「*新しい名前でのオブジェクトの保存*」を参照）。

Natural for Windows プログラムエディタを使用して、テキスト定数を 16 進 Unicode 表現に変換できます（Natural for Windows『*エディタ*』ドキュメントの「*プログラムエディタ*」セクションの「*16 進形式への変換*」を参照）。 したがって、UTF-8 ソースが望ましくないプラットフォーム用に開発している場合、Unicode 定数のすべての文字を入力し、定数のすべての文字を選択し、それらを 16 進表示に変換して、Unicode 16 進定数の "UH" 接頭辞を追加できます。 さらに、テキスト定数の文字または選択された文字範囲の上にマウスポインタを置くと、対応する 16 進 Unicode 表現がツールヒントに表示されます。

バイトオーダマーク（BOM）は、データ文字列の先頭の文字コード "U+FEFF" で構成されます。その場所で、主にマークがないプレーンテキストファイルの、バイト順およびエンコード形式を定義する署名として使用できます。 Windows では、バイトオーダマークは一部のエディタ（メモ帳など）によって、UTF-8 ファイルをマークするために使用されます。 Natural for Windows エディタでは、UTF-8 バイトオーダマークはオブジェクトを読み込むときに認識されます。 それまでにオブジェクトに他のエンコードが定義されていない場合は、Natural によって UTF-8 と解釈され、オブジェクトが保存されるときに、UTF-8 がそのオブジェクトのエンコードとして保存されます。 この場合、バイトオーダマークは削除されます。

6 Natural アプリケーションの入力／出力処理

このchapterでは、次のトピックについて説明します。

Unicode データの表示および入力

Unicode データを表示または入力する場合は、次の点に注意してください。

- ローカル環境で Natural for Windows を使用する場合、Natural 出力ウィンドウですべての Unicode 文字を表示および入力できます。
- リモート環境で Natural for Windows を使用する場合（SPoD）、すべての Unicode 文字を表示および入力するには、Web I/O インターフェイス（下記を参照）が必要です。

  端末エミュレーションまたは 3270/3279 のようなメインフレーム端末を経由して Natural を実行する場合、ページはデフォルトコードページに変換されてから表示されます。デフォルトコードページに含まれていないすべての文字は、置換文字で置き換えられます。 同様に、入力はコードページフォーマットでのみ可能であり、Unicode フォーマットに変換されてから U フォーマットフィールドに割り当てられます。 置換文字は ICU 変換テーブルによって定義されると考える必要があります。 この文字によっては、端末エミュレーションでガーベッジが表示される場合があります。 UNIX および OpenVMS プラットフォームでは、プロファイルパラメータ SUBCHAR を設定することによって、この置換文字を変更できます。 ただし、デフォルトコードページに含まれていない文字を表示するときは、Web I/O インターフェイスを使用することを強くお勧めします。 リモート Windows セッションを実行しているときは、どのような場合にも Web I/O インターフェイスが使用されます。

  コードページ指向のメインフレーム端末では、適切なコードページを選択することが重要です。 Natural のデフォルトコードページ、端末のコードページ、さらには端末で使用されるフォントによって、特定の文字を正しく表示する機能が決定されます。
- Natural for UNIX、Natural for OpenVMS、または Natural for Mainframes でアプリケーションを実行するときは、下記の「*Web I/O インターフェイスクライアント*」を参照してください。

Notes:

1. Windows で Unicode 対応の出力インターフェイスを使用している場合でも、現在選択しているフォントによってサポートされている Unicode 文字のみが表示されます。
2. Unicode データは 3270 端末では表示できません。

Web I/O インターフェイスクライアント

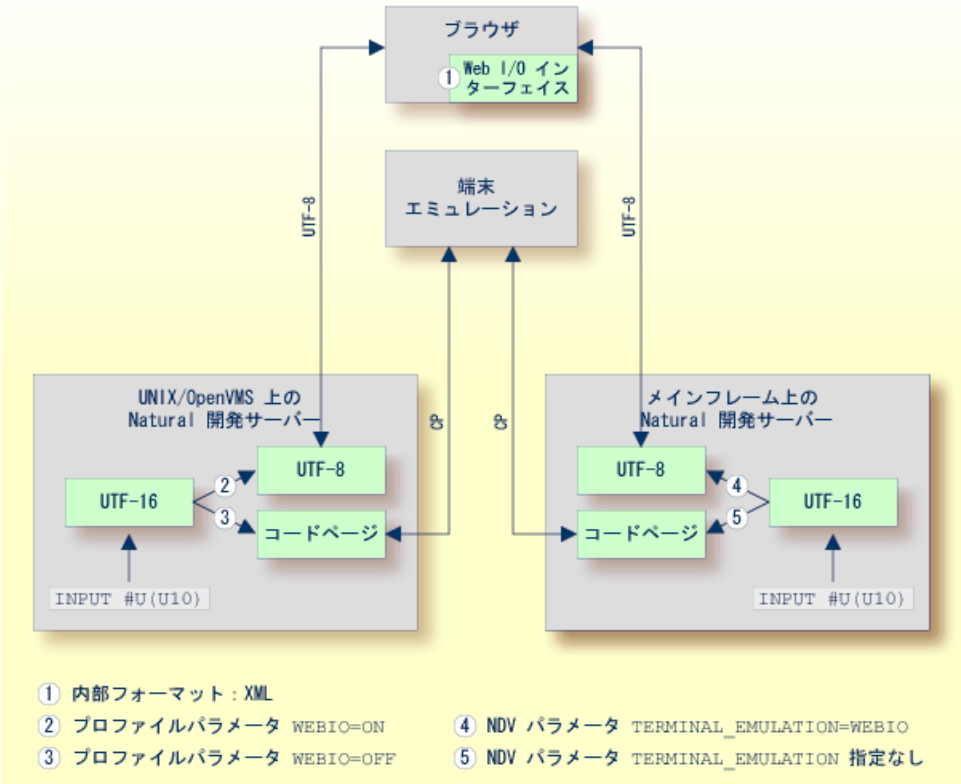
Web I/O インターフェイスクライアントは、Unicode 文字を含む非 GUI 情報を表示するために使用されます。 次の環境で使用できます。

- SPoD 環境
- ランタイム環境

SPoD 環境

Natural for Windows を使用し、リモート環境で Natural スタジオを操作しているときに（SPoD）、Web I/O インターフェイスクライアントが表示されます。 リモート UNIX、OpenVMS、またはメインフレーム環境では、Unicode 対応ではない端末エミュレーションウィンドウの代わりに表示されます。または、リモート Windows 環境では、出力ウィンドウの代わりに表示されます。

次の図に、UNIX、OpenVMS、およびメインフレームでの、Unicode アプリケーションと Natural 開発サーバー（NDV）の SPoD 環境を示します。



Web I/O インターフェイスクライアントを呼び出せるように、Natural 開発サーバーを次のように設定する必要があります。

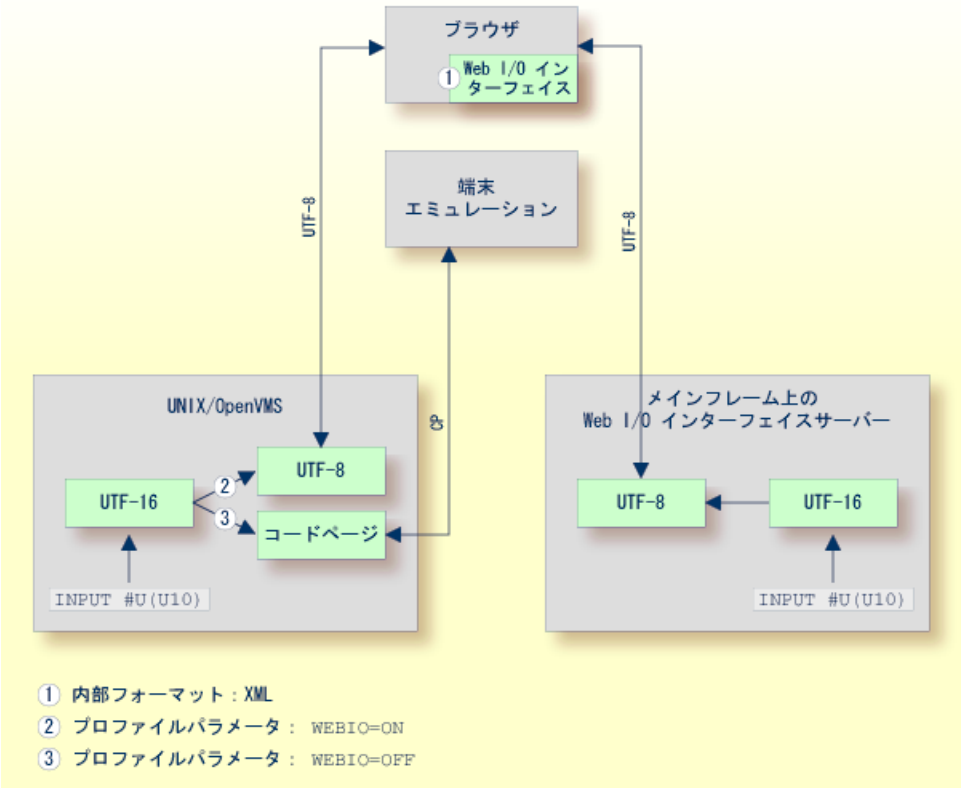
- **UNIX と OpenVMS**
  リモート UNIX または OpenVMS 環境で Web I/O インターフェイスクライアントを使用する場合は、NDV サーバーでプロファイルパラメータ WEBIO を "ON" に設定する必要があります。Natural for UNIX または Natural for OpenVMS ドキュメントの「*コンフィグレーションユーティリティ*」を参照してください。
- メインフレーム
  リモートメインフレーム環境で Web I/O インターフェイスクライアントを使用する場合は、NDV サーバーで NDV コンフィグレーションパラメータ TERMINAL_EMULATION を "WEBIO" に設定する必要があります。詳細については、『Natural Development Server』ドキュメントの「*NDV Configuration Parameters*」を参照してください。さらに、Web I/O 端末コンバータモジュール NATWEB が Natural ニュークリアスにリンクされている必要があります。ユーザー画面サイズを決めるには、Natural プロファイルパラメータ TMODEL を使用できます。
- **Windows**
  リモート Windows 環境では、プロファイルパラメータ WEBIO の設定に関係なく、Web I/O インターフェイスクライアントが常に使用されます。

Natural for Windows ドキュメントの一部である『*SPoD を使用したリモート開発*』の「*Web I/O インターフェイスクライアント*」も参照してください。

ランタイム環境

Natural for UNIX、Natural for OpenVMS、または Natural for Mainframes でアプリケーションを実行するときに、Web I/O インターフェイスクライアントが表示されます。J2EE サーバーまたは Internet Information Server で実行されます。

次の図に、UNIX、OpenVMS、およびメインフレームサーバーでの、Unicode アプリケーションのランタイム環境を示します。



Natural for UNIX では、Web I/O インターフェイスデーモンがインストールされ、アクティブになっている必要があります。 Natural for UNIX の『*インストール*』ドキュメントを参照してください。

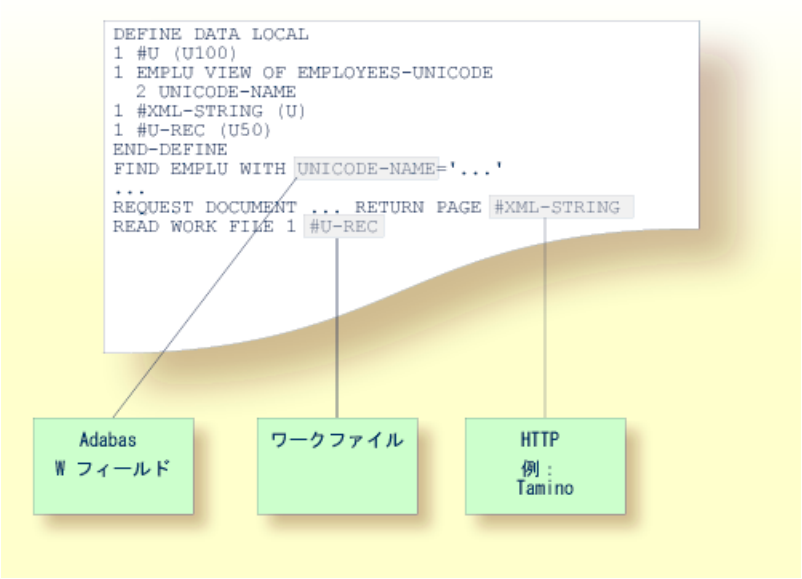
Natural for Mainframes では、Natural Web I/O インターフェイスサーバーがインストールされ、設定されている必要があります。 Natural for Mainframes の『*Natural Web I/O インターフェイスサーバー*』ドキュメントを参照してください。 さらに、Web I/O 端末コンバータモジュール `NATWEB` が Natural ニュークリアスにリンクされている必要があります。 ユーザー画面サイズを決めるには、Natural プロファイルパラメータ `TMODEL` を使用できます。

7 Unicode データストレージ

このchapterでは、次のトピックについて説明します。

Unicode データ／パラメータアクセス

次の図に、Unicode データおよびパラメータがどのようにアクセスされるかを示します。



データベース管理システムインターフェイス

Adabas データベースの Unicode データへのアクセス

Natural を使用すると、Adabas データベースのワイド文字フィールド（フォーマット W）にアクセスできます。

データ定義モジュール
: Adabas ワイド文字フィールド（W）は、Natural データフォーマット U（Unicode）にマップされます。

アクセスコンフィグレーション
: Natural は Adabas からデータを受け取り、共通のエンコードとして UTF-16 を使用してデータを Adabas に送ります。

  このエンコードは、OPRB パラメータによって指定され、オープン要求によって Adabas に送信されます。 これはワイド文字フィールドに使用され、Adabas ユーザーセッション全体を通して適用されます。

詳細については、『*プログラミングガイド*』の「*Adabas データベースのデータへのアクセス*」の「*Unicode データ*」を参照してください。

Windows、UNIX、および OpenVMS プラットフォームでのワークファイルおよび出力ファイル

以下では次のトピックについて説明します。

WRITE WORK FILE

次の情報は、ステートメント WRITE WORK FILE に適用されます。 このステートメントの詳細については、『ステートメント』ドキュメントを参照してください。

コードページデータ

次のワークファイルタイプによって、コードページデータが書き込まれます。

- ASCII および圧縮 ASCII
- Unformatted
- CSV
- Entire Connection

ワークファイルタイプおよびコードページは、コンフィグレーションユーティリティで定義される必要があります。詳細については、『*コンフィグレーションユーティリティ*』の「*ワークファイル設定*」を参照してください。

オペランド A（英数字）および U（Unicode）で定義されたすべての Natural データは、指定されたコードページに変換されます。 コードページが指定されていない場合、すべてのデータは CP パラメータで定義されたデフォルトコードページに変換されます。

Note: ワークファイルでは、書き込まれるすべての A および U オペランドデータは、コードページフォーマットです。

U オペランドデータをこれらのワークファイルに書き込み、後でこれらのワークファイルからデータを欠落させずに読み取る必要がある場合、UTF-8 をコードページとしてコンフィグレーションユーティリティで定義する必要があります。 この場合、すべての A および U オペランドデータは、UTF-8 フォーマットで書き込まれます。 ワークファイルもコードページ UTF-8 を使

用して設定されている後続の READ WORK FILE ステートメントによって、オペランド U データはデータを欠落させずに読み取られます。

Notes:

1. UTF-8 フォーマットで書き込まれたワークファイルデータは、UTF-8 をサポートするテキストエディタ（Windows プラットフォームのメモ帳など）で読み取ることができます。
2. オペランド B（バイナリ）で定義された Natural データは、コンフィグレーションユーティリティで指定されたコードページに変換されません。 これらのデータは、Natural に保存されたときのままで書き込まれ、コードページ変換は行われません。

上記のワークファイルタイプのいずれかが指定され、ワークファイルに対してコードページ UTF-8 が定義された場合、ワークファイル属性 BOM（バイトオーダマークを書き込む）および NOBOM（バイトオーダマークを書き込まない）が有効になります。 これらの属性は、コンフィグレーションユーティリティのワークファイルカテゴリで、DEFINE WORK FILE ステートメントを使用して指定できます。 ワークファイルに対してコードページ UTF-8 が定義され、ワークファイル属性 BOM が指定された場合、UTF-8 バイトオーダマーク（16 進表示：H'EFBBBF'）がワークファイルの先頭、ワークファイルデータの前に書き込まれます。

上記のワークファイルタイプ以外のワークファイルタイプがワークファイルの書き込みに使用された場合、またはワークファイルに対して UTF-8 以外のコードページが定義された場合は、ランタイム時に属性 BOM の指定は無視されます。 次の表に、ステートメント WRITE WORK FILE および READ WORK FILE の処理でのランタイム時の動作を示します。

コードページおよび属性の設定	WRITE WORK FILE	READ WORK FILE
ワークファイルにコードページ UTF-8 が指定されていません（デフォルト）。 ワークファイル属性 BOM および NOBOM は有効ではありません。	UTF-8 バイトオーダマークが書き込まれていません。 UTF-8 への変換は行われません。	UTF-8 バイトオーダマークの確認は行われません。 UTF-8 からの変換は行われません。
ワークファイルにコードページ UTF-8 が指定されています。 ワークファイル属性 BOM が指定されています。	UTF-8 バイトオーダマークが書き込まれています。 A および U フィールドは UTF-8 に変換されます。	UTF-8 バイトオーダマークを確認してください。 UTF-8 バイトオーダマークが見つかった場合、そのマークはワークファイルデータから削除されます。フィールドは UTF-8 からデフォルトコードページに変換されます。U フィールドは UTF-8 から Natural の内部ランタイム表現である UTF-16 に変換されます。
ワークファイルにコードページ UTF-8 が指定されています。	UTF-8 バイトオーダマークが書き込まれていません。	UTF-8 バイトオーダマークを確認してください。

コードページおよび属性の設定	WRITE WORK FILE	READ WORK FILE
ワークファイル属性 NOBOM（デフォルト）が指定されています。	A および U フィールドは UTF-8 に変換されます。	UTF-8 バイトオーダマークが見つかった場合、そのマークはワークファイルデータから削除されます。フィールドは UTF-8 からデフォルトコードページに変換されます。U フィールドは UTF-8 から Natural の内部ランタイム表現である UTF-16 に変換されます。

バイナリデータ

次のワークファイルタイプによって、バイナリデータ（オペランドフォーマット U の UTF-16 など）が書き込まれます。

- SAG
- Portable

オペランド A および U で定義された Natural データは、コードページに変換されません。これらのデータは、ワークファイルにバイナリフォーマットで書き込まれます。U オペランドデータの場合、これは UTF-16 で行われます。

READ WORK FILE

次の情報は、ステートメント READ WORK FILE に適用されます。このステートメントの詳細については、『ステートメント』ドキュメントを参照してください。Natural for Windows、Natural for UNIX、および Natural for OpenVMS ドキュメントで、RECORD オプションについてリストされている制限に注意してください。

コードページデータ

次のワークファイルタイプが使用されるとき、Natural U（Unicode）オペランドに読み取られるワークファイルデータは、指定されたコードページから UTF-16 に変換されます。

- ASCII および圧縮 ASCII
- Unformatted
- CSV
- Entire Connection

A（英数字）オペランドに読み取られるデータは、必要に応じて、指定されたコードページから、パラメータ CP で定義されたデフォルトコードページに変換されます。

上記のワークファイルタイプのいずれかが指定され、ワークファイルに対してコードページ UTF-8 が定義された場合、READ WORK FILE ステートメントによって、ワークファイルで UTF-8 バイトオーダマークが自動的に確認されます。ワークファイルの先頭で UTF-8 バイトオーダマーク

が見つかった場合、そのマークは削除されます。 ワークファイルから読み取られたデータは、UTF-8 からデフォルトコードページに変換されます。

データが別のワークファイルタイプから読み取られた場合、バイトオーダマークの確認は実行されず、したがってバイトオーダマークは削除されません。

ステートメント `WRITE WORK FILE` および `READ WORK FILE` の処理でのランタイム時の動作の詳細については、前のセクションの表を参照してください。

バイナリデータ

次のワークファイルタイプが使用されるとき、ワークファイルデータは変換されずに Natural オペランド A および U に読み取られます。つまり、バイナリフォーマットで読み取られます。

- SAG
- Portable

ワークファイルタイプ Portable では、オペランドフォーマット U のデータのエンディアン変換がサポートされます。

ワークファイルタイプ転送の特別な考慮事項

オペランドフォーマット U は、一般にワークファイルタイプ転送がサポートされています。 Entire Connection によって、選択されたファイルタイプの Unicode の読み取りまたは書き込みができない場合は、ランタイムエラーメッセージが表示されます。

出力ファイル

出力ファイルの Unicode データの処理は、選択された論理デバイス（LPT1～LPT31）の出力方法によって異なり、現在は GUI（Windows のみ）または TTY です。

出力方法に関係なく、データは UTF-16 フォーマットで Natural 出力サービスに渡されます。 つまり、フォーマット A フィールドデータはすでに Unicode に変換されています。

GUI 出力方法

この Windows のみの出力方法では、データは Unicode（UTF-16）フォーマットで Windows プリンタドライバに渡されます。 これは Windows での標準のデータ出力方法であるため、このデータはドライバによって常に適切に処理されます。 したがって、この出力方法は、システムコードページに含まれない文字が使用されている場合に Windows で推奨される出力方法です。

TTY 出力方法

この出力方法では、データは、デフォルトで内部（UTF-16）フォーマットからシステムコードページに変換されます。 ただし、プリンタプロファイルを使用することによって、データが代わりに UTF-8 フォーマットに変換されるか、または任意の外部コードページに追加変換される

ことを指定できます。 これらの代替手段の詳細については、『*コンフィグレーションユーティリティ*』の「*プリンタプロファイル*」を参照してください。

データをシステムコードページに変換するデフォルトの動作の理由は、現在、UTF-8フォーマットの未加工テキストデータを直接受け取ることが可能なプリンタがないことです。

8 プラットフォームの相違

Windows、UNIX、および OpenVMS プラットフォーム

Windows、UNIX、および OpenVMS プラットフォームでは、Natural は内部的に Unicode 対応です。 つまり、文字列を含む数多くの構造は、現在 Unicode フォーマットです。 例えば、現在 Natural ソースエリアは Unicode フォーマットです。 このため、Natural コードを書き込むとき、およびカタログするときに、Natural I/O および Natural 開発環境でランタイム時に Unicode データを処理できます。

最初のバージョンには、いくつかの例外があります。Natural ダイアログ（エディタおよびランタイム）は Unicode 対応ではありません。 これらのモジュールは、後のバージョンでは Unicode 対応される予定です。

Natural は内部的に Unicode 対応ですが、既存のすべてのデータには、現在はコードページフォーマットがあります。 結果として、Natural バージョン 6.2 以降で使用された場合、このようなデータはすべてコードページフォーマットから Unicode フォーマットに変換されます。 例えば、ソースがプログラムエディタで開かれた場合、コードページファイルフォーマットから Unicode ソースエリアフォーマットへの変換が実行されます。 U フォーマットを使用しない場合でも、このことには利点があります。インストールされているシステムコードページに関係なく、すべての言語固有の文字が表示されます。 ただし、正しいコードページ情報を定義するのはユーザーの責任です。 詳細については、「*既存アプリケーションの移行*」を参照してください。

Natural オブジェクトをカタログするとき、U 接頭辞を使用して定義されていないすべての定数は、対応するソースのコードページに変換されます。 ソースが UTF-8 フォーマットの場合、これらの定数はデフォルトコードページに変換されます。

Note: ほとんどの場合、Unicode データには、コードページデータよりも多くのメモリスペースが必要です。 そのため、Natural バージョン 6.2 以降では、Natural パラメータ `USIZE` を増やす必要がある場合があります。

Windows

Unicode は、ローカルな Natural for Windows 環境で完全にサポートされます。

エディタは Unicode 対応であり、使用できるすべての文字を入力できます。 ソースを保存するとき、Natural では最初にソースを元のコードページに変換しようとします。 このコードページにない文字がソースに含まれているために失敗した場合、その後の処理はパラメータ `SUTF8` の設定によって異なります。 `SUTF8` が "ON" の場合、ソースは UTF-8 フォーマットで保存されます。 `SUTF8` が "OFF" の場合、ユーザーは、ソースを元のコードページで保存するか、現在の保存をキャンセルするかを確認されます。 ソースを元のコードページで保存することをユーザーが決定した場合、見つからない文字は置換文字で置き換えられます。 また、［名前をつけて保存］ダイアログボックスで、コードページを明示的に選択できます。

プログラムエディタは、Unicode 双方向アルゴリズムをサポートするために拡張されています。

出力ウィンドウも、Unicode 対応です。 文字がキーボードで入力される場合、A フォーマットフィールドは、デフォルトコードページで使用可能な文字のみを受け入れます。

UNIX と OpenVMS

完全な Unicode サポートは、SPoD および Web I/O インターフェイスでのみ利用できます。 SPoD は、Natural ソースへの Unicode 入力に必要です。ローカルな Natural for Windows 環境について前述した内容と同じことが適用されます。 Web I/O インターフェイスは、Natural アプリケーションからの Unicode I/O に必要です。

Natural が端末エミュレーションを経由して使用される場合、すべての出力は Unicode からデフォルトコードページに変換されてから表示されます。 デフォルトコードページで使用できない文字は、デフォルトコードページの置換文字で置き換えられます。 同様の場合の入力は、デフォルトコードページに基づいてのみ可能です。

Note: UTF-8 フォーマットの Natural ソースは、Natural for UNIX または Natural for OpenVMS ネイティブエディタを使用して開くことができません。

9 既存アプリケーションの移行

このchapterでは、次のトピックについて説明します。

既存アプリケーションへの Unicode の影響

Windows、UNIX、および OpenVMS プラットフォーム

Windows、UNIX、および OpenVMS プラットフォームでは、Natural は内部的に Unicode 対応です。つまり、文字列を含む数多くの構造は、現在 Unicode フォーマットです。 例えば、現在 Natural ソースエリアは Unicode フォーマットです。 このため、コードページフォーマットでのみ使用できるデータは、内部的に Unicode フォーマットに変換されます。 このことは、例えば、Natural ソースや Natural ライブラリ名およびオブジェクト名に適用されます。 ただし、コードページから Unicode への変換、およびその逆の変換は、正しいコードページが変換に使用された場合にのみ正常に実行されます。 アプリケーションの変更ではなく再カタログのみが実行される場合でも、カタログのためにオブジェクトが Natural ソースエリアにロードされるため、コードページ情報が重要です。 すべてのオブジェクトがシステムコードページでコード化されている場合、変更は必要ありません。 オブジェクトがシステムコードページでコード化されていない場合は、追加情報について「*Windows、UNIX、および OpenVMS プラットフォームでの既存オブジェクトの移行*」を参照してください。

ほとんどの場合、内部 Unicode 構造は、より多くのメモリを必要とします。 プロファイルパラメータ `USIZE` に小さい値を定義している場合は、この値を大きくする必要がある場合があります。

Windows、UNIX、および OpenVMS プラットフォームでの既存オブジェクトの移行

Natural は、コードページ情報を複数のレベルで定義できるように拡張されています。

- Natural プロファイルパラメータ `CP` は、デフォルトの Natural コードページを定義します。
- 一部のオブジェクト（Natural ソース、Natural バッチ入力／出力ファイル、タイプ ASCII、圧縮 ASCII、Unformatted、および CSV のワークファイル）に対して、オブジェクト固有のコードページを定義できます。

オブジェクト固有のコードページもデフォルトコードページも定義されていない場合は、Natural によってシステムコードページが使用されます。

正しいコードページを自動的に識別することはできないため、必要なコードページ情報をユーザーが定義することが重要です。 次のような状況が考えられます。

ステータス	作業	対処
すべてのデータがシステムコードページで使用可能。	作業不要	対処不要。
すべてのデータを 1 つのコードページで保存。ただし、このコードページはシステムコードページとは異なる。	簡単	Natural プロファイルパラメータ CP を正しいコードページに設定する必要があります。
データは異なるコードページで使用可能。	ソースとコードページの数に依存	すべての Natural オブジェクトに正しいコードページを定義する必要があります。 ■ ソース 影響を受けるオブジェクトが少数のみの場合は、［プロパティ］ダイアログボックスを使用してコードページを変更します。 複数のオブジェクト（ライブラリ全体など）が影響を受ける場合は、FTOUCH ユーティリティを使用してコードページを変更します。 ■ バッチファイル Natural プロファイルパラメータ CPOBJIN、CPSYNIN、および CPPRINT を正しいコードページに設定します。 ■ ワークファイル コンフィグレーションユーティリティでワークファイルの正しいコードページを設定します。
1 つのオブジェクト（ソースなど）に異なるコードページが混在。	多大	オブジェクトを UTF-8 フォーマットで再書き込みする必要があります。

既存アプリケーションへの Unicode サポートの追加

U フォーマットに基づく新しいソースコードで既存アプリケーションを拡張することは、簡単です。 （A フォーマットと比較して）U フォーマットについては次のルールを考慮する必要があります。

- U 以外のフォーマットへの U の REDEFINE を避ける必要があります。文字が分割される場合があるためです。
- U フォーマットは、エンディアンに依存します。 フォーマット B と U 間で移動する場合は、これを考慮する必要があります。
- パフォーマンス上の理由から、DEFINE DATA で U を整列します（UNIX および OpenVMS でのパフォーマンスの向上）。

- U を A に移動すると文字が失われる場合があることに注意します。

既存フィールドを A フォーマットから U フォーマットに変更する場合は、次のルールを考慮する必要があります。

- 文字列の特定のエンコードを前提とするコードは、変更する必要があります（Bフィールドとの比較など）。
- フィールドのすべての REDEFINE ステートメントについて、その有効性をチェックする必要があります。
- N への REDEFINE は不可能です。つまり、予期した結果を得られません。
- データベースフィールドは Unicode に移行する必要があります（データベースでサポートされている場合）。
- フィールドの長さを変更する必要がある場合があります。A フィールドに DBCS 文字が含まれている場合、U フィールドには半分の長さが必要です。

Natural リモートプロシージャコール（RPC）の移行

プロファイルパラメータ CP の名前が CPRPC に変更されました。 以前の Natural バージョンでは、CP は、トランスポートプロトコル ACI（EntireX Broker）が使用される場合に、Entire Conversion Service（ECS）によって使用され、Natural リモートプロシージャコールにのみ適用されるコードページの名前を指定するために使用されました。

バージョン 6.2（Windows および UNIX）、バージョン 6.3（OpenVMS）、およびバージョン 4.2（メインフレーム）以降は、Natural データのデフォルトコードページを定義する新しい CP パラメータを使用できます。 Natural RPC を使用しており、以前は CP パラメータを動的に使用していた場合は、このパラメータを CPRPC に変更する必要があります。

Windows、UNIX、および OpenVMS プラットフォーム

以前のバージョンのパラメータファイルを使用する場合、何も変更する必要はありません。コンフィグレーションユーティリティによって、CP は CPRPC に自動的に移行されます。

10 特別な考慮事項と制限事項

Windows、UNIX、および OpenVMS プラットフォーム

- Natural for Windows で提供されるダイアログエディタおよびダイアログベースのランタイムは、Unicode 対応ではありません。
- Natural for UNIX および Natural for OpenVMS で提供されるエディタは、Unicode 対応ではありません。
- 250 文字を超えるフィールドに対して DL パラメータが指定された場合、最大250 文字がフィールドに表示されます。
- Natural のソース行の長さは、250 バイトを超えることはできません。 Unicode フォーマット上で機能するプログラムエディタでは、UTF-16 コード単位の数が 250 を超えないことのみがチェックされます。ただし、ソースのエンコードによっては、エンコードを UTF-16 からソースのエンコードに変換するときに、行の長さが増える場合があります。 例えば、UTF-8 エンコードでは、中国語 1 文字に最大で 4 バイトが必要です。この場合、エラーが表示され、変更は保存されません。
- UNIX および OpenVMS の場合、Unicode はランタイムに Web I/O インターフェイスでのみサポートされます。 アプリケーションが端末エミュレーションまたは xterm で実行され、Unicode 文字列が表示される場合、不正な結果となる場合があります。
- コードページと Unicode の間でいくつかの変換を実行する必要があるため、以前の Natural バージョンと比較してパフォーマンスは低下しています。

11 双方向言語サポート

このchapterでは、Windows、UNIX、および OpenVMS プラットフォームでの Natural による双方向言語のサポート方法について説明します。

アラビア語やヘブライ語などの一部の言語は右から左（RTL）に記述されますが、英語やドイツ語などの大部分の言語は左から右（LTR）に記述されます。 左から右と右から左の両方が含まれるテキストは、双方向テキストと呼ばれます。

Naturalは、双方向言語を基本的にサポートします。 Windowsでは、このサポートは、Naturalデフォルトコードページと Windows システムコードページの両方が双方向コードページとして定義されている場合に有効です。 Naturalで特定のコードページが定義されていない場合は、双方向の Windows システムコードページが定義されていれば十分です。 UNIX および OpenVMS では、双方向言語のサポートは、Naturalデフォルトコードページが双方向コードページである場合に有効です。

Natural プログラムの出力は、端末コマンド %V およびセッションパラメータ PM を使用して制御できます。

端末コマンド %V は、画面方向を定義します。 画面方向が右から左の場合、現在のウィンドウのレイアウトはミラーリングされます。つまり、すべてのウィンドウコンポーネントまたはフィールドの基点は右上隅になります。 画面方向は、%VON を使用して右から左に変更され、%VOFF を使用して左から右に戻されます。

プロファイルパラメータ PM は、デフォルトの画面方向を定義します。 PM が "R"（リセット）に設定されている場合、デフォルトの画面方向は左から右です。 PM が "I"（逆）に設定されている場合、デフォルトの画面方向は右から左です。

セッションパラメータ PM では、フィールドの方向が反対になります。「フィールドの方向を反対にした場合」 の結果は、PM パラメータが使用されるステートメントおよびプラットフォームによって異なります。 PM パラメータが MOVE ステートメントで使用された場合、フィールドの内容は単純に反対になります。つまり、最初の文字が最後の文字になるなどです。結果はフィールドの文字に依存しません。 末尾の空白が削除されてから、フィールドは反対にされます。

例えば、次のようなプログラムがあります。

```
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
```

次の出力を生成します。

```
TEST1 program    TEST2 margorp
```

"margorp" は "program" が反対になったものです。

PM パラメータが INPUT や DISPLAY などの IO ステートメントに対して使用される場合、その結果はより複雑です。 この場合、フィールド方向は画面方向に基づきます。

- 画面方向が左から右であり、PM=I がフィールドに適用される場合、フィールド方向は右から左に変わります。
- 画面方向が右から左であり、PM=I がフィールドに適用される場合、フィールド方向は左から右に変わります。

Windows では、「フィールド方向を反対にすること」は、フィールドの文字が単純に反対になることを意味しません。 Windows の複雑な双方向アルゴリズムが適用されます。詳細については、Microsoft Windows のドキュメントを参照してください。 一方、UNIX および OpenVMS では、フィールドの文字は再ソートされません。単純に反対になるだけです。

次の例では、変数 TEST に割り当てられた文字は、次の順序で入力されています。

a b c ｳｲｵ 1 2 3

次に、Windowsのプログラム例を示します。プログラムエディタで入力するときに、定数の文字はすでに再ソートされています。

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

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
```

このプログラムでは、Windows 上に次の 2 つの画面が生成されます。

```
TEST abc 123 גבש
TEST          123 גבש  abc
```

および

```
                                                                    123 גבש
 abc TEST
                                                              abc 123 גבש
   TEST
```

次に、UNIXおよびOpenVMSのプログラム例を示します。文字が同じ順序で入力された場合、文字は単純にキーイング順に表示されるため、プログラムは次のように表示されます。

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

このプログラムでは、UNIX および OpenVMS 上に次の 2 つの画面が生成されます。

```
TEST abc אבג 123
TEST          321 גבא  cba
```

および

```
                                                                    321 גבא
 cba TSET
                                                              abc אבג 123
   TSET
```

マップエディタには、［マップの反転］コマンドがあり、双方向フィールドを持つマップを簡単に処理できます。 このコマンドによって、現在のマップの表示方向が変更されます。 フィールドの位置は変わらず、表示のみが変更されます。 Windows では、このコマンドは現在のマップにのみ影響します。 UNIX および OpenVMS では、以降のすべてのマップが反対に表示されるように、フラグが設定されます。次の［**Reverse Map**］コマンドによって、元の状況が復元されます。

Windows では、ダイアログの出力を同様に制御できます。ダイアログ自体とほとんどのダイアログコントロールの両方に RTL 属性があります。 ダイアログの RTL 属性をオンにすると、ダイアログの画面方向は右から左になります。 他のコントロールの RTL 属性をオンにすると、これらのコントロールの方向は右から左になります。

プロファイルパラメータ PM によって、新しいダイアログの RTL 属性のデフォルト設定が定義されます。 PM が "R"（リセット）に設定されている場合、RTL 属性はデフォルトでオンになりません。 PM が "I"（逆）に設定されている場合、RTL 属性はデフォルトでオンになります。 ダイアログの新しく作成されるコントロールの RTL 属性のデフォルト設定は、ダイアログの RTL 属性の設定から派生します。

ダイアログにすでにコントロールがあるときにダイアログの RTL 属性が変更された場合は、コントロールの RTL 属性も変更するかどうかを確認するダイアログが表示されます。

Windows で双方向言語を使用している場合は、"GUI" が推奨される出力方法です。 出力方法 "GUI" を使用すると、出力されるページに、画面上に表示されるウィンドウと同じレイアウトが表示されます。 フィールド文字のソートは同一です。 出力方法 "TTY" を使用すると、ほとんどの場合、出力されるレイアウトは画面ウィンドウのレイアウトとは異なります。これは、フィールド文字が論理順に出力されるためです。 方向が右から左へのフィールドの場合、すべての文字が単純に反対になります。つまり、最初の文字が最後の文字になるなどです。

12 ダブルバイト文字サポート

ほとんどの東アジア言語では、コードページ文字列内の言語固有の文字（Naturalフォーマット A）は2バイト（いわゆるダブルバイト文字セット）で表され、ASCII文字（メインフレーム上のEBCDIC）は1バイトで表されます。 そのため、コードページ文字列は、1バイトと2バイトという異なる長さの文字で構成されます。

Naturalは、ダブルバイト文字セットを基本的にサポートします。 Windowsでは、このサポートは、Naturalデフォルトコードページと Windowsシステムコードページの両方がダブルバイトコードページとして定義されている場合に有効です。 Naturalで特定のコードページが定義されていない場合は、ダブルバイトのWindowsシステムコードページが定義されていれば十分です。 UNIXおよびOpenVMSでは、ダブルバイト文字セットのサポートは、Naturalデフォルトコードページがダブルバイトコードページである場合に有効です。 メインフレームでは、プロファイルパラメータ `CP` が EBCDIC MBCSコードページ（IBM-942など）に設定されている必要があります。

ダブルバイト文字セットのサポートが有効な場合、Naturalでは、すべての文字列操作について、ダブルバイト文字セットが1つのユニットとして処理されることが保証されます。 このことは、文字列の意味を保持するために不可欠です。

フォーマット Aの変数の操作の後（例えば、`SUBSTRING` オプションを使用してサブストリングを抽出した後）でダブルバイト文字セットの先頭または末尾の1バイトが残された場合、このバイトは空白文字で置き換えられます。

次の例では、コードページ "Shift_JIS" が選択されています。 変数 `#A` には、4文字で構成された文字列が含まれています。 最初と最後の文字は、コードページ "Shift_JIS" でバイトシーケンス "0x8282" によって表されるダブルバイト文字セット "FULL WIDTH LATIN SMALL LETTER B" です。 2番目と3番目の文字は、1バイト "0x61" によって表されるシングルバイト文字 "LATIN SMALL LETTER A" です。 したがって、文字列全体の16進表示は "0x828261618282" です。

```
DEFINE DATA LOCAL
 1   #A   (A10)
END-DEFINE

#A := 'ｂaaｂ'

WRITE #A #A (EM=H(6))
EXAMINE #A FOR PATTERN 'Ｂ' REPLACE 'a'
WRITE #A #A (EM=H(6))

END
```

ダブルバイト文字セットがサポートされない場合、上記のプログラムの出力は次のとおりです。

```
Page         1                              07-02-07    17:22:09

ｂaaｂ     828261618282
Ｂaｂ      826161828220
```

これは、文字"ｂ"（コードページ"Shift_JIS"で"0x8282"）が1つのユニットとして扱われなかった結果です。 この文字の末尾のバイトおよび次の文字 "a"（"0x61"）が、誤ってダブルバイト文字セット"Ｂ"（コードページ "Shift_JIS" で "0x8261"）として解釈されています。

ダブルバイト文字セットがサポートされる場合、プログラムの出力は予期したとおりになります。

```
Page         1                              07-02-07    17:22:09

ｂaaｂ     828261618282
ｂaaｂ     828261618282
```

13 よくある質問

このchapterでは、次のトピックについて説明します。

起動エラー「Invalid code page specified」が表示されるのはなぜですか。

プロファイルパラメータ CP で定義したコードページが、存在しないか（有効な ICU コードページについては *http://demo.icu-project.org/icu-bin/convexp*、適切な IANA 名については *http://www.iana.org/assignments/character-sets* を参照）、またはプラットフォームで無効なデフォルトコードページです（例えば、EBCDIC コードページは Windows、UNIX、または OpenVMS プラットフォームでは使用できません）。

"デフォルトコードページ" とは何ですか。

デフォルトコードページとは、プロファイルパラメータ CP によって定義されたコードページです。 CP が入力されない場合は、デフォルトコードページは現在のシステムコードページです。

どのデフォルトコードページが現在使用されていますか。

コードページから Unicode への変換、およびその逆の変換のために Natural によって現在使用されているデフォルトコードページは、システム変数 *CODEPAGE の内容を表示することによって知ることができます。

すべての Natural ソースを UTF-8 フォーマットで保存する必要がありますか。

使用する文字およびソースを保存するプラットフォームによります。 Unicode 定数を使用する場合、文字のすべての組み合わせを保存できるのは UTF-8 のみです。 ただし、16 進 UH 定数を定義して、それをコードページソースに保存することもできます。 16 進定数の短所は、定数のすべての文字の UTF-16 エンコードを知る必要があることです。 メインフレームでは、ソースの UTF-8 フォーマットは使用できません。 UNIX および OpenVMS では、UTF-8 ソースは SPoD 経由でのみ処理できます。UNIX または OpenVMS 上でローカルに処理することはできません。

どうすれば Natural コードで UTF-8 エンコードを処理できますか。

UTF-8 から UTF-16 への変換のために MOVE ENCODED ステートメントを使用します。A フォーマット変数に対してコードページ "UTF-8" を使用する必要があります。

一部の文字が正しく表示されないのはなぜですか。

正しいコードページを使用しているかどうかを確認してください。 コードページが正しい場合は、選択したフォントによって、表示する文字がサポートされているかどうかを確認してください。

Natural ソースを編集するときにエラーが発生するのはなぜですか。

ソースに対して定義されているコードページが正しくありません。 ソースの内容を Unicode に変換するときに、変換エラーが発生します。 Unicode に正常に変換されるように、ソースのエンコードを変更してください。

Natural ソースを保存するときにエラーが発生するのはなぜですか。

ソースの読み取りに使用されたコードページに変換されない文字をソースに入力しました。 これらの文字を誤って入力したか、本当にソースに保存するかを確認してください。 最初の場合には、誤った文字を削除してソースを保存します。 2 つ目の場合には、ソースを UTF-8 フォーマットで保存するか、または文字が U 定数に含まれている場合は、代わりに UH 定数を使用します。

ソースのコードページに含まれていない文字を入力していない場合は、プロファイルパラメータ SRETAIN が "OFF" に設定されているかどうかを確認します。 この設定の場合、ソースは現在のコードページで保存されます。 当該のソースが以前に別のコードページで保存された場合は、変換エラーが発生する場合があります。

Natural ソースのエンコードはどうすればわかりますか。

Natural スタジオでは、ソースノードの［プロパティ］ダイアログボックスを呼び出します。［一般］ページに、ソースのエンコードが表示されます。［エンコード］テキストボックスが空の場合、ソースに対して特定のエンコードは保存されていません。つまり、ソースを読み取るときにデフォルトのエンコードが使用されます。

Naturalスタジオのリストビューウィンドウにも、リストされているすべてのオブジェクトのエンコードが表示されます。

Natural ソースのエンコードはどうすれば変更できますか。

Natural スタジオでは、ソースノードの［プロパティ］ダイアログボックスを呼び出します。［一般］ページに、ソースのエンコードが表示されます。これが正しいエンコードではない場合は、［変更］ボタンを選択して変更できます。使用できるコードページのリストが表示され、ソースに対して正しいエンコードを選択できます。

どうすれば既存の Natural ソースを UTF-8 フォーマットに変換できますか（Windows、UNIX、および OpenVMS のみ）。

正しいコードページを使用して Natural エディタでソースを開きます。［名前をつけて保存］でソースを保存し、［名前をつけて保存］ダイアログボックスでエンコードとして UTF-8 を選択します。

文字を変換できない場合、どの置換文字が使用されますか。

変換の方向によって異なります。コードページの文字をUnicodeに変換できない場合は、Unicode置換文字 "U+FFFD" が使用されます。Unicode の文字をコードページに変換できない場合は、このコードページに対して ICU によって定義されている置換文字が使用されます。

Unicodeからデフォルトコードページへの変換の場合、Windows、UNIX、およびOpenVMSプラットフォームでは、プロファイルパラメータ SUBCHAR を設定することによって、置換文字を変更できます。

以前の Natural バージョンで UTF-8 ソースを使用できますか。

できません。以前の Natural バージョンでは、コードページ情報は認識されません。UTF-8 ソースは、現在のシステムコードページとして解釈されます。

UTF-8 フォーマットのソースをカタログするときに変換エラーが発生するのはなぜですか。

コードポイントを変換できないため、UTF-8 フォーマットの Natural ソースはカタログできません（Windows、UNIX、および OpenVMS のみ）。

UTF-8 フォーマットのソース内のすべての A 定数は、生成されるプログラムに保存するときに、デフォルトコードページに変換されます。 デフォルトコードページに含まれない文字を A 定数から削除するか、A 定数の代わりに U 定数を使用してください。

端末エミュレーションを経由して U フォーマットを表示するときに、UNIX または OpenVMS でガーベッジが表示されるのはなぜですか。

端末エミュレーションに出力が表示される前に、デフォルトコードページに含まれないすべての文字は、コードページの置換文字で置き換えられます。 ASCII コードページの場合、ICU 変換テーブルによって定義される置換文字が "0x1A" である場合があります。これは、UNIX または OpenVMS 端末で制御文字である場合があります。 I/O ステートメントで U フォーマットを使用する場合は、Web I/O インターフェイスを使用することを強くお勧めします。端末エミュレーションの使用が不可欠な場合は、置換文字（SUBCHAR）を出力可能な文字（"?" など）に変更できます。

現在の SPoD クライアントと古い SPoD サーバーを使用できますか。

はい。ただし、SPoD クライアントのコードページをサーバーソースのコードページに設定する必要があります。

「*Prerequisites for Natural Single Point of Development*」（***http://documentation.softwareag.com/natural/spod_prereq/prereq.htm***）を参照してください。

現在の SPoD サーバーと古い SPoD クライアントを使用できますか。

はい。ただし、ソースのエンコードを定義している場合は、お勧めしません。

「*Prerequisites for Natural Single Point of Development*」（***http://documentation.softwareag.com/natural/spod_prereq/prereq.htm***）を参照してください。

索引

ふ

ろ

わ