Support of Different Character Sets with NATCONV.INI

Support of Different Character Sets with *NATCONV.INI*

The settings in the configuration file *NATCONV.INI* apply to the A format. For the U format, the ICU library is used.

This chapter describes how Natural supports different character sets. It covers the following topics:

- Why is the Support of Different Character Sets Important?
- Character Sets that are Supported
- How to Use Different Character Sets

Why is the Support of Different Character Sets Important?

The support of multiple languages with different character sets represents Natural's approach towards internationalization. It can help you when using:

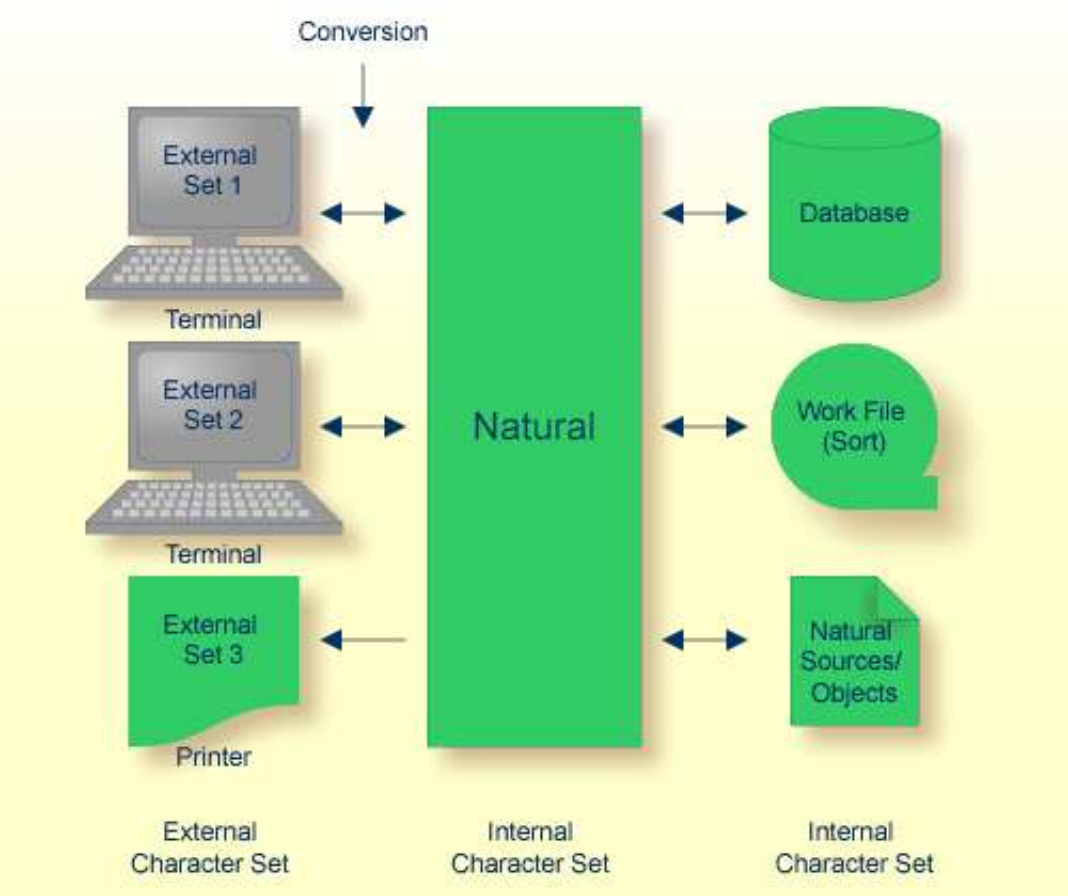
- terminals and printers with different character sets, all communicating with the same Natural environment;
- several Natural environments sharing one database and located on different platforms;
- upper-/lower-case translation of language-specific characters;
- language-specific characters in Natural identifiers, object names and library names;
- language-specific characters in an operand compared with a mask definition (see *MASK Option* in the *Programming Guide*).

Character Sets that are Supported

Natural supports any single-byte character set that conforms to the ASCII character set in the lowest seven bits.

Natural distinguishes between an internal character and several external character sets; the internal character set is used by Natural itself.

As illustrated below, conversion between the internal and an external character set is performed after the input from a terminal and before the output to a terminal or printer. There is no conversion to an external character set available for work file I/Os, database I/Os and reading/writing of Natural objects.



Internal Character Set

By default, Natural uses the internal character set "ISO8859_1". If the default character set does not meet your requirements, you can choose either one of the predefined character sets provided by Natural or any other standard character set.

Note:
Problems may occur if you run computers with different internal character sets sharing the same database, or if you try to exchange data or programming objects between such computers.

External Character Sets

You can define an external character set for any terminal and printer.

For a terminal, the name of its character set is defined by the TCS entry in the terminal database, for example: ":TCS = usascii:".

You can also use the UNIX environment variable `$NATTCHARSET` which overrides all TCS settings.

If neither a TCS entry nor the logical `NATTCHARSET` (which is set with the environment variable `$NATTCHARSET`) is defined, no conversion is performed during terminal I/O.

For a printer, the name of an external character set name can be defined in the printer profile. This is part of the global configuration file. See *Printer Profiles* in the *Overview of Configuration File Parameters* of the *Configuration Utility* documentation.

How to Use Different Character Sets

All check, translation and classification tables used by Natural to support language-specific characters reside in the configuration file *NATCONV.INI*. By default, this file is located in Natural's *etc* directory (*$NATDIR/$NATVERS/etc/*).

You can modify *NATCONV.INI* to support local or application-specific character sets.

In a standard application, *NATCONV.INI* need not and should not be modified, because this could lead to serious inconsistencies, in particular if Natural objects and database data are already present.

Modifications are necessary if you want to do any of the following:

- use an internal character set other than the default one,
- use a terminal or printer whose character set is not supported by *NATCONV.INI*,
- allow or disallow the use of certain characters in identifiers,
- support local characters when evaluating the `MASK` option.

Any modifications of *NATCONV.INI* should be well considered and carefully performed, otherwise problems might occur that are difficult to locate.

NATCONV.INI is subdivided in sections and subsections. The following sections are defined:

Section	Description
`CHARACTERSET-DEFINITION`	This section defines the name of the internal character set. The default is "ISO8859_1". If you choose a different character set, subsections for this character set must be contained in the sections described below.
`CHARACTERSET-TRANSLATION`	This section contains the tables required for the conversion between the internal character set and external character sets. If you use, for example, a terminal with an entry in `SAGtermcap` of ":TCS = ASCII_GERMAN:" and if "ISO8859_1" is used as internal character set, the following two subsections must be contained in this section: • `[ISO8859_1->ASCII_GERMAN]` • `[ASCII_GERMAN->ISO8859_1]`

Section	Description
CASE-TRANSLATION	This section contains the tables required for the conversion from upper-case to lower-case which is performed when one of the following is specified: • the terminal command %U, • the field attribute AD=T, • the statement EXAMINE TRANSLATE. This conversion is done within the internal character set. If the internal character set is, for example, "ISO8859_5", the following two subsections must be contained in this section: • [ISO8859_5->UPPER] • [ISO8859_5->LOWER]
IDENTIFIER-VALIDATION	This section contains the tables required for the validation of identifiers (that is, user-defined variables in source programs), object names and library names. It contains a subsection for each defined internal character set. The special characters "#" (for non-database variables), "+" (for application-independent variables), "@" (for SQL and Adabas null or length indicators) and "&" (for dynamic source generation) can be redefined in this section. In addition, the set of valid first and subsequent characters for identifiers, object names and library names can be modified. **Note:** When extending the set of valid characters for object names with values greater than "x7f" (decimal 127), the sorting sequence of the objects (for example, during a LIST * command) may not be in the numerical order.
CHARACTER-CLASSIFICATION	This section contains the tables required for the classification of characters, which, for example, are used when evaluating the MASK option. It contains a subsection for each defined internal character set.

The section CHARACTERSET-DEFINITION and each subsection contain lines which describe how characters are to be converted and which characters are related with which attributes. These lines are represented as follows:

```
line         ::=  key = value
key          ::=  name_key | range_key
name_key     ::=  keyword{ CHARS }
keyword      ::=  INTERNAL-CHARACTERSET | NON-DB-VARI | DYNAMIC-SOURCE |
                    GLOBAL-VARI | FIRST-CHAR | SUBSEQUENT-CHAR |
                    LIB-FIRST-CHAR | LIB-SUBSEQUENT-CHAR | ALTERNATE-CARET
                    ISASCII | ISALPHA | ISALNUM | ISDIGIT | ISXDIGIT |
                    ISLOWER | ISUPPER | ISCNTRL | ISPRINT | ISPUNCT |
```

```
                    ISGRAPH | ISSPACE
range_key     ::=  hexnum | hexnum-hexnum
value         ::=  val {, val }
val           ::=  hexnum | hexnum-hexnum
hexnum        ::=  xhexdigithexdigit | xhexdigithexdigit
```

Notes:

1. If the `range_key` variable is specified on the left-hand side, the number of values specified on the right-hand side must correspond to the number of values specified in the key range, unless only one value is specified on the right-hand side, which is then assigned to each element of the key range.
2. When the `name_key` variable is specified on the left-hand side and the corresponding list of character codes does not fit in one line, it can be continued on the next line by specifying "name_key =" again. You must not start the lines with leading blanks or tabulators.

Examples of Valid Lines

x00-x1f = x00	All characters between "x00" and "x1f" are converted to "x00".
x00-x7f = x00-x7f	All characters between "x00" and "x7f" are not converted.
x00-x08 = x00,x01-x07,x00	The characters "x00" and "x08" are converted to "x00" and characters between "x01" and "x07" are not converted.
ISALPHA = x41-x5a,x61-x7a,xc0-xd6,xd8 ISALPHA = xd9-xf6,xf8-xff	The attribute `ISALPHA` is assigned to all characters specified in these two lines.

Examples of Invalid Lines

x41 = ’A’	All characters must be specified in hexadecimal format.
0x00-0x1f = 0x00	Hexadecimal values have to be specified in either of the following ways: x*digitdigit* X*digitdigit*
x00-x0f = x00,x01	The number of specified values does not correspond to the number of elements in the key range.