CHAPTER 12 NUMBERS

As stated in *Common Lisp: the Language*, the values of named constants are implementation-dependent. The section that follows lists the limits for the Xerox Common Lisp implementation of the constants described by Steele.

12.10. Implementation Parameters

```
most-positive-fixnum ⇒ 65535
most-negative-fixnum ⇒ -65536
most-positive-short-float ⇒ 3.4028235E+38
least-positive-short-float ⇒ 1.40129847E-45
least-negative-short-float ⇒ -1.1754945E-38
most-negative-short-float ⇒ -3.4028235E+38
most-positive-single-float ⇒ 3.4028235E+38
least-positive-single-float ⇒ 1.40129847E-45
least-negative-single-float ⇒ -1.1754945E-38
most-negative-single-float ⇒ -3.4028235E+38
most-positive-double-float ⇒ 3.4028235E+38
least-positive-double-float ⇒ 1.40129847E-45
least-negative-double-float ⇒ -1.1754945E-38
most-negative-double-float ⇒ -3.4028235E+38
most-positive-long-float ⇒ 3.4028235E+38
least-positive-long-float ⇒ 1.40129847E-45
least-negative-long-float ⇒ -1.1754945E-38
most-negative-long-float ⇒ -3.4028235E+38
short-float-epsilon ⇒ 1.1920929E-7
single-float-epsilon ⇒ 1.1920929E-7
double-float-epsilon ⇒ 1.1920929E-7
long-float-epsilon ⇒ 1.1920929E-7
```

```
short-float-negative-epsilon ⇒ 5.9604645E-8
single-float-negative-epsilon ⇒ 5.9604645E-8
double-float-negative-epsilon ⇒ 5.9604645E-8
long-float-negative-epsilon ⇒ 5.9604645E-8
```

CHAPTER 13 CHARACTERS

13.1. Character Attributes

Characters in Xerox Common Lisp follow the Xerox NS Character Code Standard. Character codes are 16-bit quantities, partitioned into 8 bits of character set and 8 bits of character within the set. The value of the constant `char-code-limit` is 65536. Characters are an immediate data type; i.e., they consume no storage.

Xerox Common Lisp supports neither font nor bits attributes. Hence, the values of the constants `char-font-limit` and `char-bits-limit` are both one . The functionality of font and bits attributes are achieved instead through graphics programming conventions and the use of a larger character space.

Characters do not themselves have "font" attributes; however, streams have a notion of a current font, and some programs attach fonts to larger entities, such as strings or subranges of a file. In addition, the Xerox Character Standard encodes in the character itself some information that other implementations associate with a font. For example, a lower-case beta (β) is a distinct character (in the Greek character set), rather than being a lower-case b with a Greek font attribute. This distinct character can be rendered in an assortment of fonts.

The use for which "bits" attributes were originally intended was to represent different keyboard keystrokes (e.g., meta-hyper-A). The equivalent functionality is achieved in Xerox Common Lisp by assigning codes from other character sets to keystrokes using the key action table. Xerox Common Lisp follows the convention that characters typed with the Meta key depressed are in character set 1. Most of the extra function keys on the keyboard are in character set 2.

```
char-code-limit ⇒65536

char-font-limit ⇒ 1

char-bits-limit ⇒ 1
```

13.2. Predicates on Characters

XCL considers `graphic-char-p` to be true for exactly those characters in the space that the Xerox Character Code Standard calls "graphic" or "rendering". This space consists of characters whose character set component is zero or in one of the octal ranges [41, 176] or [241, 376], and whose character byte is in one of the octal ranges [40, 176] or [241, 376]. In particular, all of the normal ASCII printing characters are in the range [40, 176] in character set 0, and hence are graphic. Not all graphic characters are necessarily defined or have a rendering in any particular font.

The character sets 1 thru 40 and 177 thru 240 (octal) are in the range that the Xerox Character Code Standard calls "control characters". Of these character sets, only sets 1 and 2 have any assigned meaning in this release of Xerox Common Lisp.

13.4. Character Conversions

The names of most non-graphic characters are of the form *cset-char*, where *cset* is a character set name (e.g., Greek) or its octal representation (0 to 376), and *char* is the octal representation of the character within the set (0 to 376). The ASCII control characters (codes 1 through 32 octal) have names of the form "↑letter", e.g., `(cl:char-name (cl:code-char 2))` ⇒ `"↑B"`. Some well-known characters, including those documented in *Common Lisp: the Language*, have more interesting names, which are registered in the association list `il:characternames`. Users are free to add to this list, but should beware of reading characters with new names in a system that has only the default `il:characternames`. Names of well-known character sets are registered on the list `il:charactersetnames`.

Graphic characters have no name (`char-name` returns `nil`) and print as themselves.

You can input characters using the same syntax, or you can give the character within the character set a name. For example, `#\1-A` and `#\1-101` are the same character (capital A in character set 1, or the character obtained by typing Meta-A). Lowercase beta can be typed #\Greek-142 or #\46-142; being a graphic character, it always prints as #\β. However, you cannot use the "names" of lowercase letters, because Common Lisp reads case-insensitively. Thus, it could not distinguish #\Greek-B from #\Greek-b.

13.5. Character Control-Bit Functions

The constants char-control-bit, char-meta-bit, char-super-bit, and char-hyper-bit have the value zero (0), since non-zero bits attributes are not supported in XCL. In addition, the functions char-bit and set-char-bit exist but signal an error when called.

[This page intentionally left blank]

CHAPTER 15 LISTS

15.1. Conses

Xerox Common Lisp assumes trees are non-circular. Therefore, passing circular lists to these functions results in undefined actions (likely to be stack overflows or infinite loops).

15.2. Lists

`pushnew` uses the same keywords as `adjoin`, not the same ones as the standard sequence operands.

15.5. Using Lists as Sets

It is an error to hand these functions lists which are not true sets.

[This page intentionally left blank]