IEEE Standards Interpretations for IEEE Std 1003.2™-1992 IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX®)--Part 2: Shell and Utilities

IEEE Standards Department, Copyrights and Permissions, 445 Hoes Lane, Piscataway, New Jersey 08855-1331, USA

Interpretation Request #147
Topic: yytext **Relevant Sections:** A.2.7.1

POSIX.2-1992, Page 697, Section A.2.7.1, lines 371-375 state: “Implementations shall accept either of the following two mutually exclusive declarations in the Definitions section: %array Declare the type of yytext to be a null-terminated character array. %pointer Declare the type of yytext to be a pointer to a null- terminated character string.” Several years ago we internationalized our C compiler and utilities and yytext was changed from a “char” array to unsigned. This was to support the input of Latin (ISO8859) characters. According to ANSI/ISO C Standard (ISO9899:1990), whether a “char” is signed or not is implementation defined and our C compiler defines it as signed. Hence it was necessary to modify the yytext output as “unsigned char” arrays in order to support full 8-bit characters without sign extension.

200|484 1 21:55:55|TP Start 520|484 1 3218 1 1|Assertion #53 (C): Test the %pointer semantic and yytext ty i ng 520|484 1 3218 1 1|the following lines are grep’ed from lex.yy.c 520|484 1 3218 1 2|# define ECHO fprintf(yyout, “%s”,yytext) 520|484 1 3218 1 3|extern unsigned char yytextarr[]; 520|484 1 3218 1 4|extern unsigned char *yytext; 520|484 1 3218 1 5|yytext=yytextarr; 520|484 1 3218 1 6|unsigned char yytextuc[YYLMAX lex.ex lex.sh lex.yy.c lex_in 5 3_1 lex_out_53_1 makefile out.stderr out.stdout tet1.3206 tet_deletes tet_lock t et_stderr tet_tests tet_tmpfiles tet_tmpres tet_xres sizeof(wchar_t)]; 520|484 1 3218 1 7|wchar_t yytextarr[YYLMAX]; 520|484 1 3218 1 8|wchar_t *yytext; 520|484 1 3218 1 9|wchar_t yytextarr[1]; 520|484 1 3218 1 10|wchar_t yytext[YYLMAX]; 520|484 1 3218 1 11|unsigned char yytextuc; 520|484 1 3218 1 12|unsigned char yytextarr[YYLMAX]; 520|484 1 3218 1 13|unsigned char *yytext; 520|484 1 3218 1 14|unsigned char yytextarr[1]; 520|484 1 3218 1 15|char yytext[YYLMAX]; 520|484 1 3218 1 16|unsigned char yytext[YYLMAX]; 520|484 1

3218 1 17|yylastch = yytextuc; 520|484 1 3218 1 18|yylastch = (unsigned char *) yytext; 520|484 1 3218 1 19|yylastch = yytext; 520|484 1 3218 1 20|yylastch = yytextu520|484 1 3218 1 22|yylastch = yytext+yy eng; 520|484 1 3218 1 23|yylenguc = yylastch-yytextuc+1; 520|484 1 3218 1 24|yytextuc[yylenguc] = 0; 520|484 1 3218 1 25|yyleng = yylastch-(unsigned char*)yytext+1; 520|484 1 3218 1 26|yyleng = yylastch- yytext+1; 520|484 1 3218 1 27|yytext[yyleng] = 0; 520|484 1 3218 1 28|sprint(yytextuc); 520|484 1 3218 1 29|sprint(yytext); 520|484 1 3218 1 30|if (yytextuc[0] == 0 /Mail /SCT /bin /debug.out /dev /doL e tc /export /home /lib /lost+found /net /opt /sbin /sh.ragaa /stand /tmp /tmp_m t /usr /var && feof(yyin) */) 520|484 1 3218 1 31|if (yytext[0] == 0 /Mail /SCT /bin /debug.out /dev /doL /e c /export /home /lib /lost+found /net /opt /sbin /sh.ragaa /stand /tmp /tmp_mnt / usr /var && feof(yyin) */) 520|484 1 3218 1 32|yyprevious = yytextuc[0] = input(); 520|484 1 3218 1 33|yyprevious = yytext[0] = input(); 520|484 1 3218 1 34|noBytes = MultiByte(yytextuc[0],sec,third,fourth); 520|484 1 3218 1 35|noBytes = MultiByte(yytext[0],sec,third,fourth); 520|484 1 3218 1 36|output(yyprevious=yytextuc[0]=sec); 520|484 1 3218 1 37|output(yyprevious=yytext[0]=sec); 520|484 1 3218 1 38|output(yyprevious=yytextuc[0]=sec); 520|484 1 3218 1 39|output(yyprevious=yytextuc[0]=third); 520|484 1 3218 1 40|output(yyprevious=yytext[0]=sec); 520|484 1 3218 1 41|output(yyprevious=yytext[0]=third); 520|484 1 3218 1 42|output(yyprevious=yytextuc[0]=sec); 520|484 1 3218 1 43|output(yyprevious=yytextuc[0]=third); 520|484 1 3218 1 44|output(yyprevious=yytextuc[0]=fourth); 520|520|484 1 3218 1 46|output(yyprevious=yytext[0]=third); 520|484 1 3218 1 47|output(yyprevious=yytext[0]=fourth); 520|484 1 3218 1 48|yylastch=yytextuc; 520|484 1 3218 1 49|yylastch=(unsigned char*)yytext; 520|484 1 3218 1 50|yylastch=yytext; 520|484 1 3218 1 51|inspect journal to ensure that yytext is declared as a poi t er to type char 220|484 1 102 21:56:01|INSPECT 410|484 53 1 21:56:01|IC End484 1 3218 1 45|output(yyprevious=yytext[0]=sec);c yylenguc; 520|484 1 3218 1 21|yylastch = (unsigned char *)yytext+yyleng;

We believe this change to be correct and that it does not violate the POSIX.2 standard; however, we received a differing opinion. We request an official interpretation on the matter of whether the POSIX.2 standard disallows the "unsigned char" array definition. Thank you for your attention to this matter.

Interpretation Response

POSIX.2, page 697, Section A.2.7.1, lines 371-375, clearly references null-terminated character array. The C standard Section 6.1.2.5 describing types, clearly states the three types, char, signed-char, unsigned-char, are collectively called character types, therefore, POSIX.2 clearly does not specify whether yytext is an array of char, signed char or unsigned char, only that it is one of these three. The standard clearly states the acceptable types for a character array, and conforming implementations must conform to this.

Rationale for Interpretation

None.