



inter-office memo

Harvey

TO: DISTRIBUTION

FROM: Carol Shaw

CBl

DATE: 4/18/80

SUBJECT: BASIC BNF

Here is the BNF for Atari BASIC. It is more complete than the simplified format used in the BASIC manual. However, certain oddities of the language are not included, such as the fact that STR\$ and CHR\$ can be used only once in a logical comparison.

BASIC BNF by Carol Shaw 3/27/80 1:41 PM

DK1:BNFC.LST

This is the BNF for BASIC as of the above date. The notation used here is similar to that in the BASIC manual with the exception that ! is used to indicate that one and only one item should be chosen from within the curly braces. Square brackets [] indicate an optional item. Square brackets with the item inside followed by an ellipsis [...] indicates that the item may be repeated zero or more times. Non-terminals are enclosed in less than and greater than signs < >. Terminals are indicated by capital letters and should be typed as is (including '.' for abbreviations). Semantics are indicated by appropriate names for the non-terminals and by ranges given at the right.

<program> ::= <deferred statement> ! <program> <deferred statement>

<deferred statement> ::= <lineno constant> <compound statement> CR

<immediate statement> ::= <compound statement> CR

<compound statement> ::= <statement> | <compound statement> : <statement>

<statement> ::= BYE | B. |
CLOAD | CLOA. |
{ CLOSE | CL. } #<iocb> |
CLR |
{ COLOR | C. } <iexp16> | 0-3 or 0-255
CONT | CON. |
CSAVE | CS. |
{ ENTER | E. } <filespec> |
{ DATA | D. } <data string> [, <data string> ...] |
DEQ | DE. |
{ DIM | DI. } { <string id> (<iexp15>) | <mvar> } [, { <string id> (<iexp15>) | <mvar> } ...] |
DOS | DO. |
{ DRAWTO | DR. } <x>, <y> |
END | E. |
{ FOR | F. } <avar> = <aexp> TO <aexp> [STEP <aexp>] |
{ GET | GE. } #<iocb>, <avar> |
{ GOSUB | GOS. } <lineno> |
{ GOTO | GO TO | G. } <lineno> |
{ GRAPHICS | GR. } <iexp16> | 0-B, +16 (not split), +32 (no clear)
IF <aexp> THEN <lineno constant> | <compound statement> |
{ INPUT | I. } [#<iocb> {, | ;}] <avar> | <string id> [, <avar> | <string id> ...] |
{ LET | LE. }] <avar> = <aexp> |
{ LET | LE. }] <mvar> = <aexp> |
{ LET | LE. }] <svar> = <sexp> |
{ LIST | L. } [<lineno> [, <lineno>]] |
{ LIST | L. } <filespec> [, <lineno> [, <lineno>]] |
{ LOAD | LO. } <filespec> |
{ LOCATE | LOC. } <x>, <y>, <avar> |
{ LPRINT | LP. } [<exp>] [{, | ;}] [<exp>] ... |
NEW |
{ NEXT | N. } <avar> |
{ NOTE | NO. } #<iocb>, <sector>, <byte within sector> |
ON <iexp> GOTO <lineno> [, <lineno> ...] |
ON <iexp> GOSUB <lineno> [, <lineno> ...] |
{ OPEN | O. } #<iocb>, <icax1>, <icax2>, <filespec> |
{ PLOT | PL. } <x>, <y> |
{ POINT | P. } #<iocb>, <sector>, <byte within sector> |
{ POKE | POK. } <iexp16>, <byte> |
POP |
{ POSITION | POS. } <x>, <y> |
{ PRINT | PR. | ? } { #<iocb> | [<exp>] } [{, | ;}] [<exp>] ... |
{ PUT | PU. } #<iocb>, <iexp16> |
RAD |
{ READ | REA. } { <avar> | <string id> } [, { <avar> | <string id> } ...] |
{ REM | R. | <space> } [<character> ...] |
{ RESTORE | REB. } [<lineno>] |
RETURN | RET. |

```

{ RUN | RU. } [ <filespec> ] |
{ SAVE | S. } <filespec> |
{ SETCOLOR | SE. } <color reg> , <color> , <lum> |
{ SOUND | SO. } <sound reg> , <frequency> , <control> , <volume> |
{ STATUS | ST. } #<iocb> , <avar> |
STOP | STO. |
{ TRAP | T. } <iexp16> | <lineno> or 4000 to clear
{ XIO | X. } <icom> , #<iocb> , <icax1> , <icax2> , <filespecs>

```

```

<function> ::= ABS(<aexp>) |
ADR(<svar>) |
ASC(<sexp>) |
ATN(<aexp>) |
CLOG(<aexp>) |
COS(<aexp>) |
EXP(<aexp>) |
FRE(<aexp>) |
INT(<aexp>) |
LEN(<sexp>) |
LOG(<aexp>) |
PADDLE(<byte>) | 0-7
PEEK(<iexp16>) |
PTRIO(<byte>) | 0-7
RND(<aexp>) |
SQN(<aexp>) |
SIN(<aexp>) |
SQR(<aexp>) |
STICK(<byte>) | 0-3
STRIO(<byte>) | 0-3
USR(<iexp16> [, <iexp16>... ] ) |
VAL(<sexp>)

```

```

<function> ::= CHR$( <iexp16> ) |
STR$( <aexp> )

```

```

<exp> ::= <aexp> | <sexp>

```

```

<aexp> ::= <term1> | <aexp> OR <term1>
<term1> ::= <term2> | <term1> AND <term2> | NOT <term2>
<term2> ::= <term3> | <term2> <lop> <term3>
<term3> ::= <term4> | <term3> <caddop> <term4>
<term4> ::= <factor1> | <term4> <mulop> <factor1>
<factor1> ::= <factor2> | <factor1> ^ <factor2>
<factor2> ::= <factor3> | <sign> <factor3>
<factor3> ::= <sexp> <lop> <sexp> |
<unsigned constant> |
(<aexp>) | <avar> | <avar> | <function>

```

```

<sexp> ::= <string> | <svar> | <sfunction>

```

```

<lop> ::= < | > | <= | >= | = | <> ' <' and '>' are not metasymbols here
<caddop> ::= + | -
<mulop> ::= * | /
<sign> ::= + | -

```

```

<avar> ::= <identifier>
<svar> ::= <matrix id>[(<iexp15>[(<iexp15>)] )]
<svar> ::= <string id>[(<iexp15>[(<iexp15>)] )]

```

```

<string id> ::= <identifier>#
<matrix id> ::= <identifier>

```

```

<identifier> ::= <letter> [ (<letter> | <digit> ) ... ]
<letter> ::= ... AIB:CIDIEIFIGIHIIJIKILIMINIOPIQIRISITUIVIVIXIVIZ

```

```

<signed constant> ::= <sign> <unsigned constant>
<unsigned constant> ::= <digit string> [ . [ <digit string> ] ] [ E [ <sign> ] <digit> [ <digit> ] ]
    . <digit string> [ E [ <sign> ] <digit> [ <digit> ] ]
<digit string> ::= <digit> [ <digit> ... ]
<digit> ::= 0111213141516171819
<string> ::= "[ <character> ... ]"
<data string> ::= [ <data character> ... ]
<character> ::= <data character> . ]
<data character> ::= Any ASCII char other than a CR or a comma
CR ::= <this is the symbol for a carriage return>
<space> ::= <this is the symbol for a space>

<filespec> ::= <sexp>
<filespecs> ::= <sexp>
<iexp16> ::= <aexp> positive -- rounded to integer 0-65535
<iexp15> ::= <aexp> 0-32767
<byte> ::= <aexp> 0-255
<iocb> ::= <byte> device # 1-7
<icom> ::= <iexp15> 0-255
<icax1> ::= <iexp15> 0-255
<icax2> ::= <iexp15> 0-255
<lineno> ::= <iexp15> 0-32767
<lineno constant> ::= <unsigned constant> 0-32767
<x> ::= <iexp16> 0-?
<y> ::= <iexp16> 0-?
<color reg> ::= <iexp16> 0-4
<color> ::= <iexp16> 0-15
<lum> ::= <iexp16> 0-14 (even)
<sound reg> ::= <iexp16> 0-3
<frequency> ::= <iexp16> 0-255
<control> ::= <iexp16> 0-15
<volume> ::= <iexp16> 0-15
<sector> ::= <avar> 0-?
<byte within sector> ::= <avar> 0-?

```