APPLE

APPLE II® BASIC

QUICK REFERENCE GUIDE

by Gilbert Held

1807 175 YEARS OF PUBLISHING 1982 WILEY QUICK REFERENCE GUIDES

2.95

REFERENCE GUIDE NOTATIONS AND FORMAT CONVENTIONS

A standard scheme for presenting the general format of BASIC language statements is employed in this reference guide. The capitalization, punctuation and other conventions are listed below:

[ ] Brackets indicate that the enclosed items are optional. Brackets do not appear in the actual statements.

{ } Braces indicate that a choice of one of the enclosed items is to be made. Braces do not appear in the actual statements.

. . . Ellipses indicate that the preceding item may be repeated. Ellipses do not appear in the actual statements.

Italics Italics indicate generic terms. The programmer must supply the actual value or wording required. See Generic Terms and Abbreviations.

Line number A line number is implied for all BASIC language statements.

Punctuation All punctuation characters, including commas, semicolons, colons, quotation marks and parentheses, must appear as indicated.

UPPER CASE Upper case letters and words must appear exactly as indicated.

APPLE BASIC VERSIONS

Applesoft—Comprehensive BASIC which permits numeric values with fractions. Line numbers must be between 0 and 63999.

Integer BASIC—A version of BASIC which limits calculated values to a range of 32767 to −32767, and does not permit decimal fractions. Line numbers must be between 0 and 32767.

VIDEO PROMPT CHARACTERS

Character	Language
*	Monitor
>	Integer BASIC
]	Applesoft

GENERIC TERMS AND ABREVIATIONS

Arg—Argument.

Bn—A byte number.

col—Low resolution graphics column number. 0 ≤ numeric expression ≤ 39.

colh—High resolution graphics column number. 0 ≤ numeric expression ≤ 279.

const—Any numeric or string constant.

Dn—A disk drive number. Must be specified as DØ or D1.

expr—Any valid Apple II expression.

expr$—Any valid Apple II string constant, variable or expression.

exprnm—Any numeric constant, variable or expression.

filename—A disk file name.

format—Format of BASIC language statement common to both Applesoft and Integer BASIC.

format$_{AS}$—Applesoft BASIC format of BASIC language statement.

format$_{IB}$—Integer BASIC format of BASIC language statement.

length—Integer ranging from Ø through 32767 decimal.

line—A BASIC program line number.

line$_i$—The *i*th line number in a BASIC program.

Ln—Length in bytes. Integer ranging in value from 0 to 32767 decimal.

memadr—The memory address referenced by a numeric expression,variable or constant. -65535_{10} ≤ range ≤ 65535_{10}.

memloc—the memory location specified by an integer constant. 0_{10} ≤ range ≤ 65535_{10}

msg—The message denoted by quotes which forms a string.

nvar—Numeric variable.

row—Low resolution graphics row number. 0 ≤ numeric expression ≤ 47.

rowh—High resolution graphics row number. 0 ≤ numeric expression ≤ 191

Rn—Specified field of a file.

slot—predefined areas into which circuit boards plug.

Sn—Slot number for input or output. 0 ≤ n ≤ 7

sub—Subscript.

var—Numeric or string variable in Integer BASIC. Numeric, integer or string variable in Applesoft.

varnm—A numeric variable name.

var(sub)—Any subscripted numeric variable in Integer BASIC. Any subscripted integer, numeric or string variable in Applesoft.

Vn—A disk volume number. 0 ≤ n ≤ 255

VARIABLE NAMING CONVENTIONS

Applesoft format F{SL} where:

First character must be a letter.
Second character is optional and may be a letter or digit.
Last character denotes variable type: $ for string variable, % for integer variable, otherwise real variable.
Name length must be 1, 2 or 3 characters.

Integer BASIC format F{SL} where:

First character must be a letter.
Second character to last character may be letter(s) or digit(s).
Last character must be a $ for string variable.
Name length may be 1 to 100 characters.

ARITHMETIC OPERATORS

OPERATION	INTEGER BASIC	APPLESOFT
ARITHMETIC		
Exponentation	^	^
Unary Minus	–	–
Multiplication	*	*
Division	/	/
Division Remainder	MOD	Not Available
Addition	+	+
Subtraction	–	–
RELATIONAL		
Equal	=	=
Not Equal	#	< > or > <
Less than	<	<
Greater than	>	>
Less than or equal	< =	< = or = <
Greater than or equal	> =	> = or = >
BOOLEAN		
Logical complement	NOT	NOT
Logical AND	AND	AND
Logical OR	OR	OR

SYSTEM COMMANDS

AUTO—Integer BASIC command that sets the automatic line numbering mode.

format$_{IB}$: AUTO *line* [, *increment*]

CLR—Integer BASIC command that assigns zero values to all variables and array elements.

format$_{IB}$: CLR

CON—Integer BASIC command that causes a program to resume execution at the next instruction following a halt.

format$_{IB}$: CON

CONT—Applesoft command that causes a program to resume execution at the next instruction following a halt.

format$_{AS}$: CONT

DEL—Eliminates the specified program lines.

format: DEL $line_1$, $line_2$

FP—Puts the Apple II into Applesoft.

format: FP[,D*n*] [,S*n*] [,V*n*]

INT—Puts the Apple II into Integer BASIC.

format: INT

LIST—Displays all or a specified portion of the program currently in memory.

format: LIST $line_1$ [, $line_2$]

format$_{AS}$: LIST $\left\{ \begin{matrix} line_1 \, [\{ \, , \, \}] \\ [line_1] \{ \, , \, \} \, line_2 \end{matrix} \right\}$

MAN—Ends the automatic line numbering mode in Integer BASIC.

format$_{IB}$: MAN

MON—Permits the display of disk commands and/or data traffic on the video monitor.

format: MON [C] [,I] [,O]

where: C is specified to display disk commands
I is specified to display data flow from disk to Apple II
O is specified to display data flow from Apple II to the disk

NEW—Deletes the current program and all variables from memory in the Apple II.

format: NEW

NOMON—Ends the video display of disk commands and/or data flow initiated by MON command.

format: NOMON

RUN—Executes the current program in memory. If a line number is specified, execution commences at that number, otherwise execution begins at the lowest numbered line in the program.

format: RUN [*line*]

SAVE—Saves the program currently in memory on cassette or on disk. See Input/Output control for formats.

BASIC LANGUAGE STATEMENTS

BRANCHING

CALL—Results in a branch to a machine language subroutine at a specified memory location.

format: CALL *memadr*

GOSUB—Results in a branch to the indicated *line* number. A return statement causes a branch back to the instruction following the GOSUB.

format: GOSUB *line*

GOTO—Causes an unconditional branch to the indicated *line* number or the numeric expression in Integer BASIC.

format: GOTO *line*

format:$_{IB}$: GOTO *exprnm*

IF-THEN—Causes a branch or the execution of a statement to occur if the indicated expression is true.

format: IF *expr* THEN *statement* [:*statement* . . .]

IF *expr* { THEN / GOTO / THEN GOTO } *line*

format$_{IB}$: IF *expr* THEN *statement*
IF *expr* THEN [GOTO] *line*

ON-GOSUB—Causes a conditional Applesoft subroutine call. Branch depends upon the computed or current value of the expression.

format$_{AS}$: ON *exprnm* GOSUB *line* [, *line* . . .]

ON-GOTO—Causes a conditional Applesoft branch based upon the current or computed value of the expression.

format$_{AS}$: ON *exprnm GOTO line* [, *line* . . .]

POP—Causes the most recently executed GOSUB to be changed to a GOTO by removing the return location for the last GOSUB statement executed.

format: POP

RETURN—Results in a program branch to the statement immediately following the most recently executed GOSUB statement.

format: RETURN

ERROR DETERMINATION AND CONTROL

DSP—Integer BASIC statement that displays the changing values of a predefined variable during program execution.

format$_{IB}$: DSP *var*

NO DSP—Integer BASIC statement that cancels the display mode for a predefined variable.

format$_{IB}$: NO DSP *var*

NO TRACE—Turns off the program execution TRACE initiated by a TRACE statement.

format: NO TRACE

ONERR GOTO—Applesoft statement that causes a branch to the *line* number indicated if an error occurs in the program.

format$_{AS}$: ONERR GOTO *line*

RESUME—Applesoft statement that causes the program to resume at the beginning of an instruction in which an error previously occurred.

format$_{AS}$: RESUME

TRACE—Displays the line number of each statement executed.

format: TRACE

MEMORY REFERENCE

CLEAR—Applesoft statement that initializes all numeric variables and array elements to zero and assigns a null value to all strings.

format$_{AS}$: CLEAR

HIMEM—Establishes the highest location in RAM memory available for use by the BASIC program.

format: HIMEM: *exprnm*

LOMEM—Establishes the lowest location in RAM memory available for use by the BASIC program.

format: LOMEM: *exprnm*

PROCESSING STATEMENTS

DATA—Applesoft statement that creates a list of values to be assigned to variables through the use of a READ statement.

format: DATA *const* [, *const* . . .]

DEF FN—Applesoft statement that permits special purpose functions to be defined.

format: DEF FNnvar (*arg*) = *exprnm*

DIM—Reserves space in main memory for an array or string.

format$_{AS}$: DIM var(*sub*[,*sub* . . .]) [,*var*(*sub*[,*sub* . . .]) . . .]

format$_{IB}$: DIM *var*(*sub*) [,*var*(*sub*) . . .]

END—Causes the program to terminate.

format: END

FOR—Initiates a loop that repeats execution of all instructions bounded by the NEXT statement until the automatically incremented variable attains the value $exprnm_2$.

format: FOR *varnm* = $exprnm_1$, TO $exprnm_2$ [STEP $exprnm_3$]

LET—Assigns a value to the specified variable.

format: [LET] *var* = *expr*

NEXT—Terminates the loop initiated by a FOR statement

format: NEXT *varnm* [,*varnm* . . .]

READ—Applesoft statement that assigns values from DATA statements to variables.

format$_{AS}$: READ *var* [,*var* . . .]

REM—Nonexecutable statement that permits remarks to be placed in the program.

format: REM *remark*

RESTORE—Applesoft statement that resets the DATA list pointer to the beginning of the list.

format$_{AS}$: RESTORE

STOP—Applesoft statement that causes the program to halt execution.

format$_{AS}$: STOP

WAIT—Applesoft statement that halts a program until a specified memory location attains a defined value.

format$_{AS}$: WAIT *memadr, exprnm,* [,*exprnm*]

INPUT/OUTPUT CONTROL

GENERAL

GET—Applesoft statement that receives one character from the keyboard without displaying the character.

format$_{AS}$: GET *var*

IN#—Selects the peripheral *slot* from which subsequent input will occur.

format: IN# *slot*

INPUT—Optionally displays a *prompt message* and then accepts input data, assigning values to the variables listed.

format: INPUT ["*prompt message*";] *var* [, *var* . . .]

PDL—Function that returns the current value of the game control device.

format: PDL (*exprnm*)

PR#—Selects the peripheral *slot* which will receive subsequent output.

format: PR# *slot*

PRINT—Outputs values to the display or to other designated devices.

format: [*const*, *expr*, *expr$*, *exprnm*, *varnm*, *var*(*sub*)]

TAB—Integer BASIC statement that positions the cursor to the specified column at the current line.

format$_{IB}$: TAB *col*

CASSETTE CONTROL

LOAD—Loads the program on cassette into memory.

format: LOAD

RECALL—Applesoft statement that retrieves a numeric array from cassette storage.

format$_{AS}$: RECALL *varnm*

SAVE—Saves the program currently in memory on cassette storage.

format: SAVE

SHLOAD—Applesoft statement that loads a high-resolution graphics shape table from cassette storage.

format$_{AS}$: SHLOAD

STORE—Applesoft statement that saves a specified array on cassette storage.

format$_{AS}$: STORE *varnm*

JOHN WILEY & SONS
605 Third Avenue, New York, N.Y. 10158
New York • Chichester • Brisbane • Toronto • Singapore

DISK CONTROL

APPEND—Opens a disk file, positioning the file pointer at the end of the file.

*format: APPEND *filename* [, D*n*] [, S*n*] [,V*n*]

BLOAD—Loads a binary disk file starting at the memory location specified.

*format: BLOAD *filename* [,A*memloc*] [,D*n*] [,S*n*] [,V*n*]

BRUN—Retrieves, loads and executes a machine language (binary) program.

*format: BRUN *filename* [, A*memloc*] [,D*n*] [,S*n*] [,V*n*]

BSAVE—Saves a *length* of Apple II's memory in binary on designated disk file.

*format: BSAVE *filename*, A*memloc*, L *length* [, D*n*] [,S*n*] [,V*n*]

CATALOG—Provides a list of all files on the specified disk.

*format: CATALOG [, D*n*] [, S*n*]

CHAIN—Integer BASIC statement that loads and executes a program from disk.

*format$_{IB}$: CHAIN *filename* [, D*n*] [, S*n*] [,V*n*]

CLOSE—Closes all open disk files or a specified open disk file.

*format: CLOSE [*filename*]

DELETE—Erases designated file from the disk.

*format: DELETE *filename* [, D*n*] [, S*n*] [, V*n*]

EXEC—Permits designated text file to control Apple II.

*format: EXEC *filename* [, R*n*] [, D*n*] [, S*n*] [,V*n*]

FP—Retrieves Applesoft from disk.

format$_{AS}$: FP [, D*n*] [, S*n*] [,V*n*]

INIT—Initializes disk by saving program in memory on disk under designated *filename*.

*format: INIT *filename* [, D*n*] [, S*n*] [,V*n*]

LOAD—Loads program with designated *filename* from disk.

*format: LOAD *filename* [, D*n*] [, S*n*] [,V*n*]

LOCK—Protects a disk file from deletion until it is unlocked.

*format: LOCK *filename* [, D*n*] [, S*n*] [,V*n*]

MAXFILES—Permits up to 16 disk files to be open.

*format: MAXFILES *limit*

OPEN—Prepares a disk file for access.

*format: OPEN *filename* [, D*n*] [, S*n*] [,V*n*]

POSITION—Positions the disk file pointer R*n* fields in front of its current position.

*format: POSITION *filename* [, R*n*]

PR#—Selects designated peripheral slot to receive output.

*format: PR# slot

READ—Designates a disk file from which subsequent INPUT and GET commands receive data.

*format: READ *filename* [, R*n*] [, B*n*]

RENAME—Changes the name of a disk file without changing its contents.

*format: RENAME *filename*$_1$, *filename*$_2$ [, D*n*] [, S*n*] [,V*n*]

RUN—Loads and executes a program residing on disk storage.

*format: RUN *filename* [, D*n*] [, S*n*] [,V*n*]

SAVE—Permanently stores program on disk with designated *filename*.

*format: SAVE *filename* [, D*n*] [, S*n*] [,V*n*]

UNLOCK—Permits file to be changed or deleted by removing its locked status.

*format: UNLOCK *filename* [, D*n*] [, S*n*] [,V*n*]

VERIFY—Checks to insure file saved correctly on disk.

*format: VERIFY *filename* [, D*n*] [, S*n*] [,V*n*]

WRITE—Designates the disk file to which subsequent PRINT statements output data.

*format: WRITE *filename* [, R*n*] [, B*n*]

*Indicates Disk Operating System (DOS) commands which require a PRINT statement containing a CTRL-D character in the programmed mode operation.

VIDEO DISPLAY COLOR CONTROLS

COMPLEMENT COLORS

COLOR	XDRAW COLOR	COLOR	XDRAW COLOR
Black	White	Orange	Blue
White	Black	Green	Violet
Violet	Green	Blue	Orange

 ISBN 0 471-87039-0

LOW RESOLUTION GRAPHICS COLORS

COLOR	NUMBER	COLOR	NUMBER
Black	0	Brown	8
Magenta	1	Orange	9
Dark Blue	2	Gray #2	10
Purple	3	Pink	11
Dark Green	4	Light Green	12
Gray #1	5	Yellow	13
Medium Blue	6	Aqua	14
Light Blue	7	White	15

HIGH RESOLUTION GRAPHICS COLORS

COLOR	NUMBER
Black	0
Green	1
Violet	2
White	3
Black	4
Orange	5
Blue	6
White	7

VIDEO AND GRAPHICS CONTROL

COLOR—Sets the low-resolution graphics color.

format: COLOR = *exprnm*

DRAW—Applesoft statement that draws a high-resolution graphics shape on the video display.

format$_{AS}$: DRAW *exprnm* [AT *colh*, *rowh*]

FLASH—Applesoft statement that flashes the video display output.

format$_{AS}$: FLASH

GR—Puts display into low-resolution graphics mode of 40 col × 40 rows.

format: GR

HCOLOR—Applesoft statement that sets the color for plotting in the high-resolution graphics mode.

format$_{AS}$: HCOLOR = *exprnm*

HGR—Applesoft statement that puts the video display into the high-resolution graphics mode of 280 columns × 160 rows, with a four-line text window.

format$_{AS}$: HGR

HGR2—Sets the display to a full-screen, high-resolution graphics mode of 280 columns × 192 rows.

format: HGR2

HLIN—Draws a horizontal line on the display from col_1 to col_2 at the specified *row* in low-resolution graphics.

format: HLIN col_1 , col_2 AT *row*

HOME—Applesoft statement that clears the display and positions the cursor at row 1, column 1.

format$_{AS}$: HOME

HPLOT—Applesoft statement that places a dot or draws a line on the display in the high-resolution graphics mode.

formats$_{AS}$:

HPLOT *colh*, *row*
HPLOT TO *colh*, *rowh*
HPLOT $colh_1$,$rowh_1$ TO $colh_2$,$rowh_2$ [TO $colh_3$,$rowh_3$ . . .]

HTAB—Applesoft statement that moves the cursor to the specified column.

format$_{AS}$: HTAB *col*

INVERSE—Applesoft statement that puts display output into the inverse video mode.

format$_{AS}$: INVERSE

NORMAL—Applesoft statement that turns off the FLASH and INVERSE video modes.

format$_{AS}$: NORMAL

PLOT—Displays a point on the low-resolution graphics display or a colored character in the text window.

format: PLOT *col*, *row*

POS—Function that returns the column position of the cursor.

format: POS (*exprnm)*

ROT—Applesoft statement that rotates high-resolution shapes drawn by DRAW or XDRAW.

format$_{AS}$: ROT = *exprnm*

SCALE—Applesoft statement that sets the size of high-resolution shapes drawn by DRAW or XDRAW.

format$_{AS}$: SCALE = *exprnm*

SCRN—Function that returns the color code of the low-resolution graphics point at the specified coordinates.

format: SCRN (*col*, *row*)

SPC—Function that moves the cursor a specified number of positions to the right.

format: SPC (*exprnm*)

SPEED—Applesoft statement that changes the rate at which characters are output to the display.

format: SPEED *exprnm*

TEXT—Returns display to the full-screen text mode from any graphics mode.

format: TEXT

VLIN—Draws a vertical line from row_1 to row_2 in column *col* on the display in the low-resolution graphics mode.

format: VLIN row_1, row_2 AT *col*

VTAB—Positions the cursor to the line specified by *row* in the current display column.

format: VLIN *row*

XDRAW—Applesoft statement that draws shape *exprnm* at the designated location.

format$_{AS}$: XDRAW *exprnm* [AT *colh*, *rowh*]

BASIC FUNCTIONS

FUNCTION FORMAT AND DESCRIPTION

ABS (*exprnm*)—Returns the absolute value of a number.

ASC (*expr$*)—Returns the ASCII code number for the specified character.

ATN (*exprnm*)—Returns the arctangent of an angle of *exprnm* radians.

CHR$ (*exprnm*)—Returns the character (string value) of the specified ASCII code.

COS (*exprnm*)—Returns the cosine of an angle of *exprnm* radians.

EXP (*exprnm*)—Returns the base of the natural logarithm (e) raised to the specified power.

FN *varnm* (*exprnm*)—Calls the previously defined function *varnm* with the value *exprnm* assigned.

FRE (*exprnm*)—Returns the number of bytes of memory available for an Applesoft program.

INT (*exprnm*)—Returns the integer portion of a number.

LEFT$ (*expr$*, *exprnm*)—Returns the leftmost *exprnm* characters of the string *expr$*.

LEN (*expr$*)—Returns the length of the specified string.

LOG (*exprnm*)—Returns the natural logarithm of the specified number.

MID$ (*expr$*, $exprnm_1$, [$exprnm_2$])—Returns $exprnm_2$ characters from *expr$*, commencing with character $exprnm_1$.

PDL (*exprnm*)—Returns the current value of the specified game control paddle.

PEEK (*memadr*)—Returns the decimal value of a specified memory location.

POS (*exprnm*)—Returns the column position of the cursor.

RIGHT$ (*expr$*, *exprnm*)—Returns the rightmost *exprnm* characters of *expr$*.

RND (*exprnm*)—Returns a random number.

SCRN (*col*, *row*)—Returns the color code of the specified coordinates when in low-resolution graphics mode.

SGN (*exprnm*)—Returns +1 if *exprnm* is positive, −1 if negative, and 0 if its value is zero.

SIN (*exprnm*)—Returns the sine of an angle of *exprnm* radians.

SPC (*exprnm*)—Moves the cursor *exprnm* positions to the right.

SQR (*exprnm*)—Returns the square root.

STR$ (*exprnm*)—Converts a numeric value to a string.

TAB (*exprnm*)—Used with the PRINT statement to move the cursor to the specified column position.

TAN (*exprnm*)—Returns the tangent of an angle of *exprnm* radians.

USR (*exprnm*)—Causes a branch to a machine language subroutine indirectly through memory locations 10 through 12, specified by *exprnm*.

VAL (*exprnm*)—Returns the numeric value of a string.

CONTROL CHARACTER UTILIZATION

CTRL-B—Places the Apple into the version of BASIC in Read Only Memory. For the Apple II Plus this places it into Applesoft BASIC.

CTRL-C—Halts a listing.

CTRL-D—Used in conjunction with appropriate PRINT statements to direct output to, or receive input from, a peripheral device.

CTRL-G—Rings the Apple II bell.

CTRL-H—Backspace.

CTRL-J—Generates a linefeed.

CTRL-K—Boots DOS from the monitor after the appropriate slot number is typed. In Monitor Mode, substitutes the device in a given backplane slot for the Apple keyboard for data input.

CTRL-M—Generates a carriage return.

CTRL-P—When followed by appropriate slot number, boots DOS from the Monitor and Integer BASIC. In Monitor Mode, diverts all output for the screen to an Apple intelligent interface.

CTRL-S—Temporarily suspends the listing of a program until any other key is pressed.