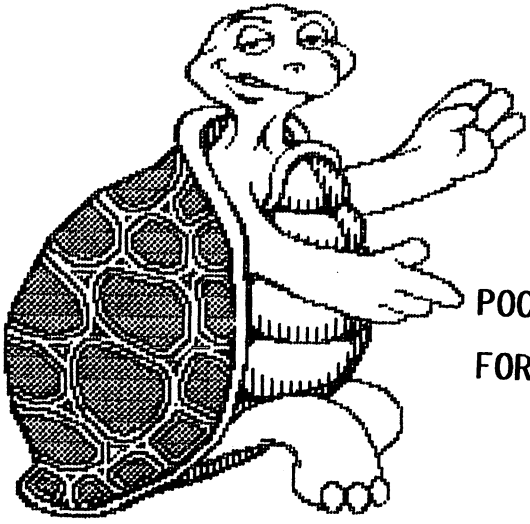
CALVIN THE COMAL TURTLE ™

PRESENTS



POCKET QUICK GUIDE FOR C64 COMAL 0.14

COMAL USERS GROUP, U.S.A., LIMITED
5501 GROVELAND TERRACE
MADISON, WI 53716-3251

C64 COMAL QUICK GUIDE

To LOAD COMAL version 0.14 from disk:

1) Computer OFF.
2) Computer ON.
3) Disk drive ON.
4) Insert diskette.
5) Enter this command:
   LOAD "BOOT*",8
6) Enter this command:
   RUN

 It is written by UniComal Aps, Denmark. It is documented by the **Commodore Approved COMAL HANDBOOK** published by Reston Publishing, Reston VA. Parts of this GUIDE are taken from COMAL TODAY newsletter, COMAL HANDBOOK, and COMMODORE 64 GRAPHICS AND SOUND WITH COMAL (to be published 1984) with permission.

WHAT IS COMAL

COMAL is an **advanced** programming language designed by Borge Christensen over 10 years ago to replace BASIC. There now are two international committees supervising its development and standardization. Len Lindsay of the COMAL Users Group (USA) and Jens Erik Jensen and Lars Lauersen of UniComal Aps are committee members. The COMAL STANDARD definition, called the COMAL KERNAL, was adopted in May 1982, and reaffirmed in March 1983. CBM COMAL follows the KERNAL.

WHY COMAL WILL REPLACE BASIC

* COMAL is **FASTER** than BASIC.
* COMAL is **STANDARDIZED.**
* COMAL is **OFFICIAL.**
* COMAL includes a **RUN-TIME COMPILER.**
* COMAL includes most of BASIC.
* COMAL has HI-RES COLOR GRAPHICS.
* COMAL has SPRITES.
* COMAL has the **'LOGO' TURTLE.**
* COMAL utilizes the function keys.
* COMAL is HELPFUL.
* COMAL is **EASY** to learn and read.
* COMAL program listings are EASY to READ.
* COMAL programs are EASY to maintain.
* COMAL includes **ADVANCED** string handling.
* COMAL is NOT SLOWED DOWN by remarks.
* Line numbers are IRRELEVANT.
* **LONG VARIABLE NAMES** do not use more memory.
* COMAL is USER EXPANDABLE.
* COMAL allows you to **MERGE** program segments.
* COMAL can read BASIC DATA FILES.
* COMAL can SORT quickly.
* COMAL is CONSISTENT and **WELL DESIGNED.**
* COMAL is **POWERFUL.**
* COMAL is **AFFORDABLE.**
* COMAL is WELL DOCUMENTED.
* COMAL is AVAILABLE **NOW.**

  COMAL is running on thousands of computers.
  Yours should be next.
* COMAL is **STRUCTURED,** including:

  CASE .. WHEN .. OTHERWISE .. ENDCASE
  IF .. THEN .. ELIF .. ELSE .. ENDIF
  REPEAT .. UNTIL
  WHILE .. ENDWHILE
  FOR .. STEP .. ENDFOR
  LOOP .. ENDLOOP
  PROC .. ENDPROC
  FUNC .. ENDFUNC

COMAL KEYWORDS: (version 0.14)

// — allows comments in a program
  //[<anything>]
ABS — gives the absolute value
  ABS(<numeric expression>)
AND — logical AND
  <expression> AND <expression>
APPEND — start at end of seq file
  OPEN [FILE] <filenum>,<filename>,APPEND
ATN — arctangent in radians
  ATN(<numeric expression>)
AUTO — automatic line numbering
  AUTO [<start line>][,<increment>]
BASIC — back into BASIC mode
  BASIC
CASE — multiple choice decisions
  CASE <control expression> [OF]
CAT — gives disk directory
  CAT [<drive number>]
CHAIN — load & run program on disk
  CHAIN <filename>
CHR$ — gives that numbers character
  CHR$(<numeric expression>)
CLOSE — closes files
  CLOSE [[FILE] <filenum>]
CLOSED — all proc or func variables local
  PROC <procname>[(params)] [CLOSED]
  FUNC <funcname>[(params)] [CLOSED]
CON — continue program execution
  CON
COS — cosine in radians
  COS(<numeric expression>)
DATA — provides data for a READ
  DATA <value>[,<value>][,...]
DEL — deletes lines
  DEL <range>
DELETE — deletes a file from disk
  DELETE <filename>
DIM — reserves/allocates string & array space
  DIM <string var> OF <max char>
  DIM <str array>(<array index>) OF <max char>
  DIM <array name>(<array index>)
DIV — division with integer answer
  <dividend> DIV <divisor>
DO — do the following statements
  DO <statements>
EDIT — lists lines without indentations
  EDIT [<range>]
ELIF — short for ELSE IF condition
  ELIF <expression> [THEN]
ELSE — alternative statements in IF structure
  ELSE
END — halt program execution
  END
ENDCASE — end of CASE structure
  ENDCASE
ENDFOR — end of FOR structure
  ENDFOR [<control variable>]
ENDFUNC — end of function
  ENDFUNC [<function name>]
ENDIF — end of IF structure
  ENDIF
ENDPROC — end of procedure
  ENDPROC [<procedure name>]
ENDWHILE — end of WHILE structure
  ENDWHILE
ENTER — merge a program segment from disk
  ENTER <filename>
EOD — End Of Data flag
  EOD
EOF — End Of File flag
  EOF(<filenum>)
ESC — stop key pressed flag
  ESC
  TRAP ESC<type>

EXEC — execute a procedure
  [EXEC] <procname>[(<actual parameter list>)]
EXP — natural log e to n
  EXP(<numeric expression>)
FALSE — predefined value = 0
  FALSE
FILE — specifies that a file is to be used
  INPUT FILE <filenum>[,<recnum>]: <var list>
  PRINT FILE <filenum>[,<recnum>]: <val list>
  READ FILE <filenum>[,<recnum>]: <var list>
  WRITE FILE <filenum>[,<recnum>]: <var list>
  OPEN [FILE] <filenum>,<filename>[,<type>]
  CLOSE [[FILE] <filenum>]
FOR — start of FOR loop structure
  FOR <var>=<start> TO <end> [STEP <step>] [DO]
FUNC — start of a multiline function

  FUNC <name>[(<params>)] [CLOSED]
GOTO — go to line with this name
  GOTO <label name>
IF — start of conditional IF structure
  IF <condition> [THEN]
  IF <condition> THEN <statement>
IN — locate position of string1 within string2
  <string1> IN <string2>
INPUT — input from keyboard or file
  INPUT [<prompt>:] <var list>
  INPUT FILE <filenum>[,<recnum>]:<var list>
INT — gives nearest integer less than or equal
  INT(<numeric expression>)
KEY$ — scans keyboard (not in PET COMAL 0.14)
  KEY$
LABEL — assigns a label name to the line
  <label name>:
LEN — gives the length of string
  LEN(<string expression>)
LET — assign value to variable
  :=
LIST — list program
  LIST [<range>] [<filename>]
LOAD — load a program from disk
  LOAD <filename>
LOG — natural logarithm of n
  LOG(<numeric expression>)
MOD — gives remainder of division (modulo)
  <dividend> MOD <divisor>
NEW — clears program from memory
  NEW
NOT — logical NOT
  NOT <condition>
NULL — does nothing (no op)
  NULL
OF — part of DIM or CASE structure
  CASE <expression> [OF]
  DIM <stringvar> OF <max char>
  DIM <stringarray>(array index) OF <max char>
OPEN — open a file
  OPEN [FILE] <filenum>,<filename>[,<type>]
OR — logical OR
  <condition> OR <condition>
ORD — returns integer representing the char
  ORD(<string expression>)
OTHERWISE — default for CASE
  OTHERWISE
OUTPUT — select output location
  SELECT [OUTPUT] <type>
PASS — passes a string to disk command chan1
  PASS <disk command>
PEEK — look at memory
  PEEK(<memory address>)
POKE — change memory location
  POKE <memory address>,<contents>
PRINT — prints items to screen/printer/file
  PRINT [FILE <filenum>:] [<items>]
  PRINT [FILE <filenum>:] USING <format>:<vars>
  (RANDOM file use: [FILE <filnum>,<recnum>:])
PROC — start of multiline procedure

  PROC <name>[(<params>)] [CLOSED]
RANDOM — random access disk file
  OPEN FILE <filenum>,<filename>,RANDOM <recln>
READ — read data from DATA line or file
  READ <var list>
  READ FILE <filenum>[,<rec num>]: <var list>
  OPEN [FILE] <filenum>,<filename>,READ
REF — param var used in reference in proc
  REF <var>
RENUM — renumber program
  RENUM [<targetstart>][,<increment>]
REPEAT — start of REPEAT structure
  REPEAT
RESTORE — reuse DATA with READ
  RESTORE
RND — random number
  RND(<num>)
  RND(<start num>:<end num>)
RUN — run program now in memory
  RUN
SAVE — record program on disk
  SAVE <filename>
SELECT — choose output location
  SELECT [OUTPUT] <type>
SGN — -1 if neg, 0 if 0, 1 if pos
  SGN(<numeric expression>)
SIN — gives sine in radians
  SIN(<numeric expression>)
SIZE — reports on memory usage (free memory)
  SIZE
SQR — gives square root
  SQR(<numeric expression>)
STATUS$ — status of disk channel
  STATUS$
STEP — increment FOR loop var by this amount
  STEP <numeric expression>
STOP — halt program execution
  STOP
SYS — transfer control to assembly language
  SYS(<memory address>)
TAB — print spaces up to specified column
  TAB(<column number>)
TAN — gives tangent in radians
  TAN(<numeric expression>)
THEN — part of IF structure
  THEN
TO — increment FOR variable start TO end
  <start num> TO <end num>
TRAP — disable stop key
  TRAP ESC<type>
TRUE — predefined value of 1
  TRUE
UNIT — specify unit (device)
  OPEN FILE <#>,<nm>,UNIT <dev>[,<sec>][,<typ>]
UNTIL — end of REPEAT loop
  UNTIL <expression>
USING — allows formatted output (not PET 0.14)
  PRINT USING <format>: <var list>
WHEN — choice in CASE structure
  WHEN <list of values>
WHILE — start of WHILE structure
  WHILE <expression> [DO] [<statement>]
WRITE — write to a file
  WRITE FILE <filenum>[,<recnum>]: <var list>
  OPEN [FILE] <filenum>,<filename>,WRITE
ZONE — tab increment
  ZONE <tab interval>
  ZONE

TURTLE GRAPHICS CHART	CBM LOGO	CBM COMAL
TURTLE CONTROL:		
Move forward length	FORWARD	FORWARD
Move backward length	BACK	BACK
Home turtle	HOME	HOME
Turn turtle left	LEFT	LEFT
Turn turtle right	RIGHT	RIGHT
Move to a point	SETXY	SETXY
Turn to specific heading	SETHEADING	SETHEADING
Make turtle visible	SHOWTURTLE	SHOWTURTLE
Make turtle invisible	HIDETURTLE	HIDETURTLE
Pen up off paper	PENUP	PENUP
Pen down on paper	PENDOWN	PENDOWN
Set pen color	PENCOLOR	PENCOLOR
Number of colors	16	16
Set size of turtle	-	TURTLESIZE
Plot a point	-	PLOT
Print text in graphics	?	PLOTTEXT
SCREEN AND COLOR CONTROL:		
Set screen window	?	FRAME
Clear graphics screen	CLEARSCREEN	CLEAR
Set to graphics mode	DRAW	SETGRAPHIC
Set to text screen	NODRAW	SETTEXT
Set background color	BACKGROUND	BACKGROUND
Set border color	-	BORDER
Fill in an area	-	FILL
Full screen mode	FULLSCREEN	FULLSCREEN
Split screen mode	SPLITSCREEN	SPLITSCREEN
FUNCTION KEYS RESULTS:		
F1	TEXT SCREEN	TEXT SCREEN
F3	SPLITSCREEN	SPLITSCREEN
F5	FULLSCREEN	FULLSCREEN

COMMODORE 64 COMAL COLORS LIST

COLOR NUMBER	COLOR NAME	CHR$	COLOR NUMBER	COLOR NAME	CHR$
0	BLACK	144	8	ORANGE	129
1	WHITE	5	9	BROWN	149
2	RED	28	10	LIGHT RED	150
3	CYAN	159	11	DARK GREY	151
4	PURPLE	156	12	MEDIUM GREY	152
5	GREEN	30	13	LIGHT GREEN	153
6	BLUE	31	14	LIGHT BLUE	154
7	YELLOW	158	15	LIGHT GREY	155

SPRITES (version 0.14)

DATACOLLISION — test for collision with data
  DATACOLLISION <sprite#>,<reset collisn flg?>
DEFINE — set up a sprite image for later use
  DEFINE <sprite definition num>,<64 byte def$>
HIDESPRITE — turn off specified sprite
  HIDESPRITE <sprite number>
IDENTIFY — assign a sprite an image
  IDENTIFY <sprite number>,<definition number>
  (note: sprite 7 is used for the turtle)
PRIORITY — does data has priority over sprite
  PRIORITY <sprite number>,<data priority?>
SPRITEBACK — set two multicolor sprite colors
  SPRITEBACK <color1>,<color2>

SPRITECOLLISION — test for sprite collision
  SPRITECOLLISION <sprite#>,<reset collsn flg?>
SPRITECOLOR — set color of sprite
  SPRITECOLOR <sprite number>,<color number>
SPRITEPOS — position sprite at x,y location
  SPRITEPOS <sprite#>,<x coord>,<y coord>
SPRITESIZE — set sprite size (expand or not)
  SPRITESIZE <sprite#>,<x expand?>,<y expand?>

HIGH RES and **TURTLE** graphics (COMAL 0.14)

BACK — move turtle backwards
  BACK <length>
BACKGROUND — set the screen background color
  BACKGROUND <color number>
BORDER — set the screen border color
  BORDER <color number>
CLEAR — clear the graphics screen
  CLEAR
DRAWTO — draws a line from current point
  DRAWTO <x coordinate>,<y coordinate>
FILL — fills in area with current color
  FILL <x coordinate>,<y coordinate>
FORWARD — move turtle forward
  FORWARD <length>
FRAME — set up a screen window
  FRAME <x0>,<x1>,<y0>,<y1>
FULLSCREEN — fullscreen graphics (f5)
  FULLSCREEN
HIDETURTLE — make the turtle invisible
  HIDETURTLE
HOME — put the turtle in its home position
  HOME
LEFT — turn turtle left
  LEFT <degrees>
MOVETO — move to specified point without line
  MOVETO <x coordinate>,<y coordinate>
PENCOLOR — sets the current turtle pen color
  PENCOLOR <color number>
PENDOWN — put pen down, turtle draws line
  PENDOWN
PENUP — pick up pen, turtle doesn't draw line
  PENUP
PLOT — plot a point in current color
  PLOT <x coordinate>,<y coordinate>
PLOTTEXT — print text on graphics screen
  PLOTTEXT <x coord>,<y coord>,<text$>
RIGHT — turn turtle right
  RIGHT <degrees>
SETGRAPHIC — turn on graphics screen
  SETGRAPHIC [<type>]
SETHEADING — set turtle heading
  SETHEADING <degree>
SETTEXT — turn on text screen (f1)
  SETTEXT
SETXY — set turtle x and y coordinates
  SETXY <x coordinate>,<y coordinate>
SHOWTURTLE — make turtle visible
  SHOWTURTLE
SPLITSCREEN — 2 text lines above graphics (f3)
  SPLITSCREEN
TURTLESIZE — set turtle size (0 to 10)
  TURTLESIZE <size>

SPECIAL INFO (COMAL version 0.14)

Line numbers allowed: 1-9999.
Identifiers up to 16 characters (unshifted alpha, digits, [,], ', <-, \).
Null input is accepted.
First time into graphics: SETGRAPHIC 0
After that simply SETGRAPHIC
RUN/STOP RESTORE keys restore default colors.
To clean up the identifier name table:
(frees up memory, removes unused identifiers)
```
 LIST "PROGRAM.L"
 NEW
 ENTER "PROGRAM.L"
```
Save a program to disk: SAVE "PROGRAM"
Load a program from disk: LOAD "PROGRAM"
List a program to printer:
```
 SELECT "LP:"
 LIST
```