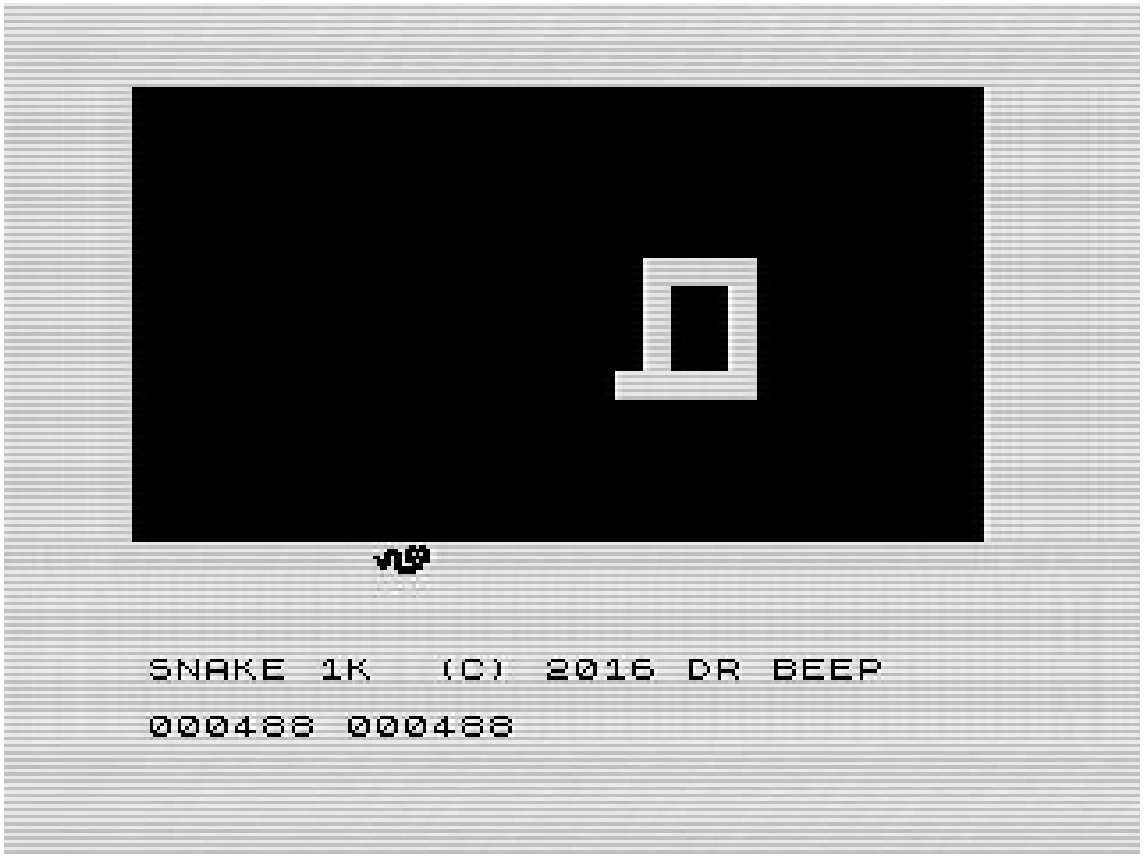
SNAKE



This game doesn't look hires. The display is in fact just space and inverted space (and a passing by snake for fun). Still without this simple displaymethod the game couldn't be coded in 1K. Besides showing a lowres growing snake on the screen the info about the snake is stored on the displaylines. This is needed to move the snake's tail. If not then not only the screen was defined, but also space for the tail. The classic game now available in 1K.

```
; Snake 1K
; Snake grows until tailbyte.
; How far can you go?
; each 32 steps grow of snake
; hires only needed to store track of tail on screen
; walking snake added for nice effects.

? * TORNADO *

            ORG  #4009              ;#4009
            DUMP 49161

            JP   init

d_file      DEFW dfile
dfcc        DEFW dfile+1
var         DEFW vars
dest        DEFW 0
eline       DEFW last
```

```
chadd       DEFW last-1
xptr        DEFW 0
stkbot      DEFW last
stkend      DEFW last               ; memory above reused for data

berg        DEFB 0
            DEFW 0
            DEFB 0
            DEFB 2
            DEFW 1

lastk       DEFB 255,255,255        ; used by ZX81

margin      DEFB 55
nxtlin      DEFW basic
            DEFB 0
            DEFB 0
flagx       DEFB 0                  ; x
strlen      DEFW 0

taddr       DEFW 3213

seed        DEFW 0
frames      DEFW 65535              ; used by ZX81
coords      DEFB 0,0
prcc        DEFB 188
sposn       DEFB 33,24
cdflag      DEFB 64

dir         DEFB %11111011
            DEFB up*256/256
            DEFB %11101111
            DEFB right*256/256
            DEFB %11111101
            DEFB down*256/256
            DEFB %11110111
            DEFB left*256/256


up          DEC  B
            LD   E,62               ; -31 = 31 prev down
            JR   movetest
left        DEC  C
            LD   E,32               ; -31 = 1 prev right
            JR   movetest
right       INC  C
            LD   E,30               ; -31 = -1 prev left
            JR   movetest
down        INC  B
            DEFB 17                 ; LD DE,"POP BC"*256+0
            NOP                     ; -31 = -31, prev up
deadtail    POP  BC                 ; only from tailbyte on stack
movetest    JR   Z,deadwall         ; wall hit
```

```
            LD   A,31
            CP   C
            JR   Z,deadwall         ; wall hit
            LD   A,18
            CP   B
            JR   NZ,playgame

deadwall    LD   HL,score-1
            LD   DE,hiscore-1
            LD   BC,7
fhigh       DEC  C
            JR   Z,start
            INC  HL
            INC  DE
            LD   A,(DE)
            CP   (HL)
            JR   Z,fhigh
            JR   NC,start
            LDIR

start       LD   A,(lastk)
            CP   %10111111
            JR   NZ,start


            LD   HL,score
            LD   B,6
erscore     LD   (HL),28
            INC  HL
            DJNZ erscore

            LD   HL,lbuf00
cls         LD   (HL),128           ; screen inverted
fnext       INC  HL                 ; also clear pointers
            LD   A,(HL)
            RLCA
            JR   NC,cls             ; data found
            RLCA
            JR   NC,cls             ; unused, no opcode
            INC  A                  ; EOline or EOscreen
            JR   NZ,fnext           ; EOLINE

            LD   BC,#0810           ; start in middle
            LD   (steps+1),A        ; reset step counter
            LD   E,B                ; first "valid" move on screen

playgame    PUSH BC
            LD   HL,lbuf00-1        ; calculate fieldaddress
ffield      INC  HL
            DEC  C
            JR   NZ,ffield          ; first calc dx, then dy
            LD   C,31               ; set dy
            DJNZ ffield
```

```
            BIT  7,(HL)            ; tailbyte?
            JR   Z,deadtail
            LD   (HL),E            ; snake on track and index

            LD   B,4               ; max tail < 1024
            LD   A,(frames)
            SUB  B
            LD   (timer+1),A       ; calculate delay

findend     DEC  BC                ; searchloop to find
            LD   A,B               ; end of tail
            OR   C
            JR   Z,deadtail        ; break tail loop, collided
            LD   A,(HL)
            SUB  31
            LD   E,A
            SBC  A,A
            LD   D,A
            ADD  HL,DE             ; calculate previous
            BIT  7,(HL)            ; end of tail reached
            JR   Z,findend         ; not end of tail yet

endfnd      AND  A                 ; undo possible carry
            SBC  HL,DE             ; final position of tail

            POP  BC                ; fetch x/y
wfr         LD   A,(frames)
timer       CP   0                 ; findend is done in delaytime
            JR   NZ,wfr            ; speeddelay

steps       LD   A,0
            INC  A
            AND  31
            LD   (steps+1),A
            LD   (snpos+1),A
            JR   Z,addscore        ; snake grows
            LD   (HL),128          ; erase tail

addscore    LD   HL,score+6        ; each step a point
            DEFB 17                ; hide reset prev. counter
tens        LD   (HL),28
            DEC  HL
            INC  (HL)
            LD   A,(HL)            ; carry to next position?
            CP   38
            JR   Z,tens

nkey        LD   A,(lastk)         ; get pressed key
            DEFB 17                ; hide previous key
lastdir     LD   A,%11101111       ; previous direction
olddir      LD   D,5               ; possible directions
            LD   HL,dir-1
dkey        DEC  D
```

```
            JR   Z,lastdir          ; no valid direction, do old
            INC  HL
            LD   E,(HL)
            CP   (HL)               ; which direction
            INC  HL
            JR   NZ,dkey
            LD   L,(HL)             ; fetch routine
            LD   (lastdir+1),A      ; set prev. direction
            JP   (HL)               ; do move


; the hires shows only lowres blocks, but is needed
; to double use the memory for display and trackkeeping

hr          LD   B,7
synch1      DJNZ synch1


            LD   HL,lbuf00+#8000
lenbuf      LD   DE,0*31            ; repaired after loading
            LD   A,#40
            LD   I,A
            LD   BC,#1008
cloop       LD   A,126              ; becomes 0 in display
            LD   R,A
            CALL #44                ; call (hl)
            DEC  C
            JR   Z,btest
            LD   A,C
            AND  7
            LD   C,A                ; 7 to 1
            NOP                     ; timing
            JP   cloop
btest       ADD  HL,DE              ; fetch next linebuffer
            DJNZ cloop

            LD   B,11
h2          DJNZ h2
            LD   A,(HL)

            LD   HL,snakeudg        ; some extra
snpos       LD   DE,#4000
            LD   B,10
setudg      LDI
            LDI
            DEC  DE
            DEC  E
            XOR  A
            CALL high+#8000
lowret      DJNZ setudg

            LD   B,7
synclow     DJNZ synclow
```

```
            LD   BC,#417           ; the lowres display
            LD   HL,dfile+#8000
            LD   A,#1E
            LD   I,A
            LD   A,#F5
            CALL #2B5

exit        CALL #292
            CALL #220
            LD   IX,hr
            JP   #2A4

high        LD   R,A
            DEFW 0,0,0,0,0,0,0,0,0,0,0,0,0,0
            RET


snakeudg    DEFB 0,54,14,127,31,107,219,127,123,119
            DEFB 123,28,51,252,1,240,0,0

n           EQU  27                ; to ZX81 Ascii

dfile       DEFB #76
            DEFB "S"-n,"N"-n,"A"-n,"K"-n,"E"-n,0,29,"K"-n,0
            DEFB 0,16,"C"-n,17,0,30,28,29,34,0
            DEFB "D"-n,"R"-n,0,"B"-n,"E"-n,"E"-n,"P"-n
            DEFB #76
            DEFB #76
score       DEFB 28,28,28,28,28,28,0
hiscore     DEFB 28,28,28,28,28,28,#76

; each line has its own buffer.
; lbuf02 and lbuf03 are created after loading

lbuf00      DEFW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
            RET

lbuf01      DEFW 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
            RET

init        LD   IX,hr             ; go to hr with 1 linebuf
            LD   SP,#4400          ; set SP
            LD   HL,#4020-1
cldata      DEC  L
            LD   (HL),0            ; clear displaydata
            JR   NZ,cldata         ; erase dataline for graphic
            LD   HL,start          ; start of game after init
            PUSH HL
            LD   HL,repairhr       ; HR routine is corrupt
            PUSH HL
            LD   HL,lenbuf+1
            PUSH HL
```

```
            LD   A,255
            LD   (vars),A          ; now end of screen marker
            LD   HL,lbuf00
            LD   DE,init           ; clear init by lbuf's
            LD   BC,62             ; 2 lines

            JP   #0A6F             ; do copy and get HL


basic       DEFB 0,0               ; only used to start program
            DEFB 0,0               ; cleared by code
            DEFB 249,212,28
            DEFB 126
            DEFB 143,0,18,0,0,0    ; 14
            DEFS 4


lbuf04      DEFW 0,0,0,0,0,0,0,0,0,0,0,0,0,0
repairhr    LD   (HL),31           ; set hr correct
            RET                    ; goto start

; built a nice startscreen
lbuf05      DEFW #8080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8080
            RET

lbuf06      DEFW #80,0,0,0,0,0,0,0,0,0,0,0,0,0,#8000
            RET

lbuf07      DEFW #0080,#8080,#0080,#8080,#0000
            DEFW #8000,#0000,#0080,#0080,#8000
            DEFW #0000,#8000,#8000,#8000,#8000
            RET

lbuf08      DEFW #0080,#0080,#0000,#0080,#0080
            DEFW #0080,#0080,#8080,#0000,#0080
            DEFW #0080,#8080,#8000,#0080,#8000
            RET

lbuf09      DEFW #0080,#8080,#0080,#0080,#0080
            DEFW #8080,#0080,#0080,#0000,#8080
            DEFW #0080,#8000,#8000,#0000,#8000
            RET

lbuf10      DEFW #0080,#0000,#0080,#0080,#0080
            DEFW #0080,#0080,#8080,#0000,#0080
            DEFW #0000,#8000,#8000,#0080,#8000
            RET

lbuf11      DEFW #0080,#8080,#0080,#0080,#0080
            DEFW #0080,#0080,#0080,#0080,#8000
```

```
            DEFW #0080,#8000,#8000,#8000,#8000
            RET

lbuf12      DEFW #0080,#0080,#0000,#0000,#0000
            DEFW #0000,#0000,#0000,#0000,#0000
            DEFW #0000,#0000,#0000,#0000,#8000
            RET

lbuf13      DEFW #0080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8000
            RET

lbuf14      DEFW #0080,#0000,#0000,#0000,#0000
            DEFW #0000,#0000,#0000,#0000,#0000
            DEFW #0000,#0000,#0000,#0000,#8000
            RET

lbuf15      DEFW #8080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8080
            DEFW #8080,#8080,#8080,#8080,#8080
            RET

vars        DEFB 128                ; becomes end of screen marker
?
last        EQU  $
```