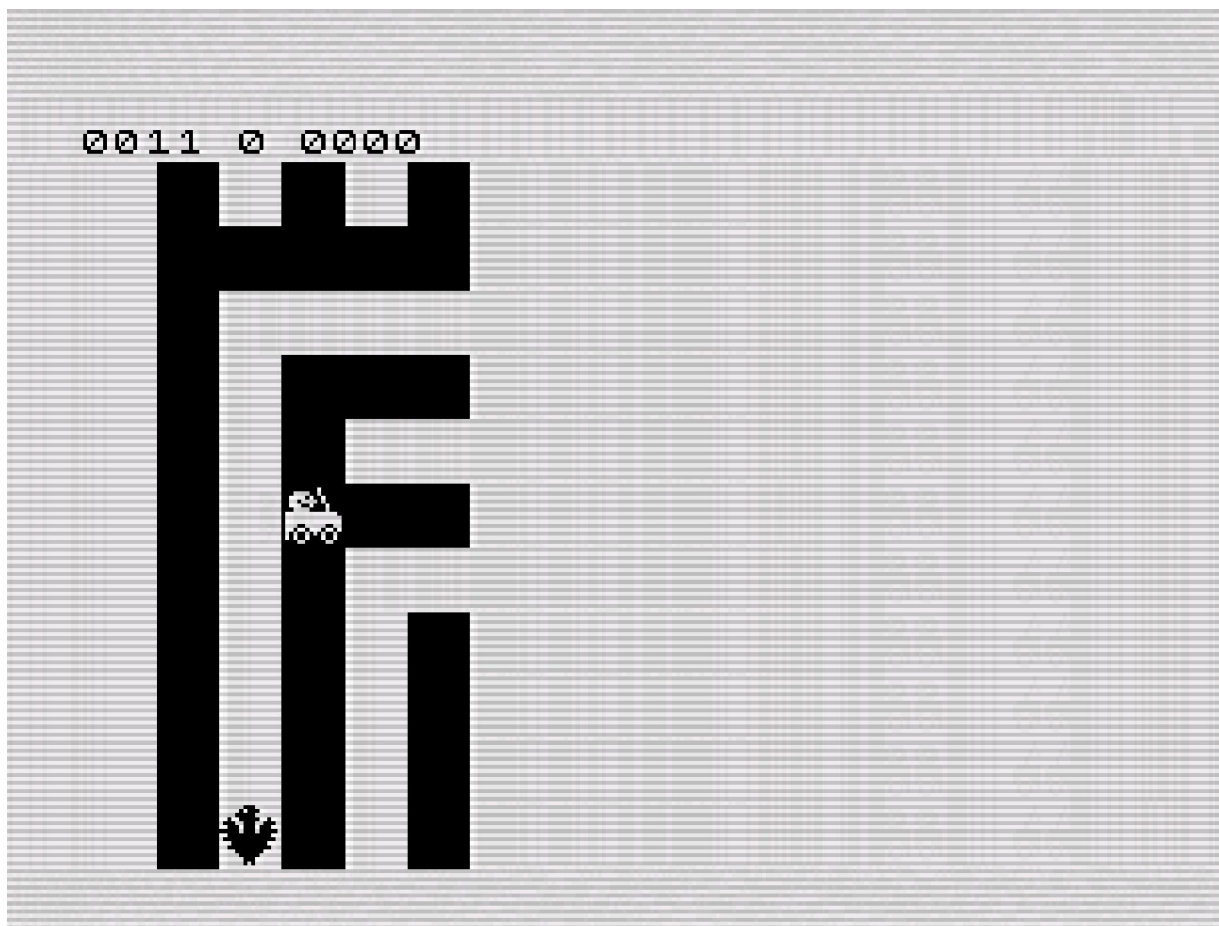


## City Escape



Can you set more than 2 small UDG's in a line during refreshing of the screen? It can, but can you alter background as well? That was the question to get this game working. Each copy will take time and therefore effect the size of the display. I managed to display 10 columns, 2 UDG's and changed background. Exactly what I need for this game. It ment I had to rotate the screenplay from the original, but I like it the way it is.

```
; City Escape
; screen is set with x=0 right.
```

```
ns      EQU  noshow*256/256
ll      EQU  linedata*256/256
```

```
? * 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
berg       DEFB 0
mem        DEFW 0                ; not needed without fp
          DEFB 128

dfsz       DEFB 2
stop       DEFW 1
lastk      DEFB 255,255,255
margin     DEFB 55
nxtlin     DEFW basic
oldppc     DEFW 0
flagx      DEFB 0
strlen     DEFW 0
taddr      DEFW 3213
seed       DEFW 0
frames     DEFW 65535
coords     DEFB 0,0
prcc       DEFB 188
sposn      DEFB 33,24
cdflag     DEFB 64

hr         LD    B,14
hr00       DJNZ  hr00

          LD    HL,lowres+#8000
          LD    BC,#208
          LD    A,#1E
          LD    I,A
          LD    A,#F5
          CALL  #2B5

;          LD    B,14
;hre       DJNZ  hre

          LD    H,udgs/256        ; highbyte
          LD    A,H
          LD    I,A
          LD    D,H              ; linedata on samehighbyte
          LD    BC,#4019         ; start of index after ldir
          EXX
          LD    IX,lbuf+#8000
          LD    DE,#0A10
          LD    HL,linedata
          LD    C,11             ; 5+1+5

repairhr   CALL  retcom          ; HR gets active after init

          CALL  #292
          CALL  #220

```

```

LD    IX,hr
JP    #2A4

cloop  ADD    A,D
LD     L,A
DEC    C
RET    Z
hrentry LD     B,E
EXX
LD     L,udgs*256/256

bloop  LD     A,(BC)           ; set first UDG on line
LD     E,A
INC    C
LDI
LDI

LD     A,(BC)           ; set second UDG on line
LD     E,A
LDI
LDI
INC    BC

EXX
DEC    B
LD     A,L
JP     (IX)

lbuf   LD     R,A
DEFW   0,0,0,0,0
JP     Z,cloop
EXX
JP     blow

blow   INC    C
INC    C
JP     bloop

init2  LD     HL,dirs
LD     DE,level
LD     BC,keycodes        ; redefinable keys
upkey  LD     A,(lastk)
INC    A
JR     NZ,upkey
PUSH   HL
downkey LD     HL,(lastk)    ; read directionkey
LD     A,L
INC    A
JR     Z,downkey
LD     A,H
LD     (BC),A             ; store part 1 directionkey

```

```

DEC BC
LD A,L
LD (BC),A ; store part 2 directionkey
POP HL
LDI ; show next direction
DEC DE
LD A,(DE)
CP 28 ; ready when level 0 shows
JR NZ,upkey ; controlkeys now defined

dead LD HL,score-1
LD DE,hiscore-1
LD BC,5
fihi DEC C
JR Z,start
INC HL
INC DE
LD A,(DE)
CP (HL)
JR Z,fihi
JR NC,start
LDIR

start LD A,(lastk)
SUB %10111111 ; wait for newline
JR NZ,start

LD HL,score-1
LD B,5
resfld INC HL
CP (HL)
JR Z,resfld
LD (HL),28
DJNZ resfld
DEC (HL)

LD A,11 ; all levels can be played
nextpig JR Z,dead ; above false

LD (pigcnt+1),A ; speed set pigeon
LD (pglpcnt+1),A ; from above 10

LD HL,level
INC (HL)

XOR A
LD (progress+1),A

LD HL,data ; erase old maze
LD B,80

```

```

cls          LD    (HL),A          ; always 0
            INC    HL
            DJNZ   cls

            LD     BC,#206
            CALL   sfield          ; startfield

xloop        LD     C,7            ; start of xloop
            LD     B,3            ; 3 vertical lines
            INC    C              ; always start on even
yloop        PUSH   BC            ; save loopcounters
rfindy       CALL   rnd           ; 0, 1 or 2
            ADD    A,A           ; 0, 2 or 4
            LD     B,A           ; the y-coordinate for a field
            CALL   field
            JR     C,rfindy       ; if already used, find new
            CALL   rnd           ; again 0, 1 or 2
            DEC    A             ; now -1, 0 or 1
            LD     E,0           ; set dx to 0
            JR     NZ,setdy       ; dy<>0, dx must be 0
            DEC    E             ; dy=0, then dx=-1
setdy        LD     D,A           ; set dy
            ADD    A,B
            CP     5
            JR     NC,rfindy      ; out of field with dy, again
            PUSH   BC            ; save coordinates
            LD     H,A           ; in between coordinate
            ADD    A,D
            LD     B,A           ; field to reach
            LD     A,C
            ADD    A,E
            LD     L,A
            LD     (xywall+1),HL ; save in between
            ADD    A,E
            LD     C,A           ; field to reach
            CALL   field
            POP    BC            ; retrieve coordinate
            JR     NC,rfindy      ; existing field reached?
            CALL   sfield        ; field from now set as used
xywall       LD     BC,0
            CALL   sfield        ; set movedirection in between
            POP    BC            ; fetch counterd
            DJNZ   yloop         ; do 3 vertical fields
            INC    C             ; test next x-field
            LD     A,C
            CP     102
            JR     C,xloop        ; do until end reached
            CALL   rnd
            ADD    A,A
            LD     B,A
            CALL   sfield        ; set exit random

```

```

        LD    BC,#306
        LD    (pigeonxy+1),BC    ; set start of pigeon
        DEC   B                  ; player startposition

gameloop  PUSH  BC                ; save xy of player

        LD    A,C
        ADD   A,5
        LD    C,A                ; start of screen is 5 further
        LD    DE,linedata
        LD    A,11                ; 11 views 5 above 5 below
xlp       LD    (lpcnt+1),A
        LD    B,5
ylp       PUSH  DE
        DEC   B
        CALL  field
        INC   B
        POP   DE
        SBC   A,A                ; make field visible
        LD    (DE),A
        INC   DE
        LD    (DE),A            ; double sized UDG and field
        INC   DE
        DJNZ  ylp
        DEC   C
lpcnt     LD    A,0
        DEC   A
        JR    NZ,xlp            ; screen now ready, add UDG

oldpigxy  LD    HL,#4000
        LD    (HL),ns            ; set to noshow

; pigeon on screen?
pigeonxy  LD    BC,0
        PUSH  BC
        CALL  field
        SBC   A,A
        LD    HL,basic-1
        CP    (HL)
        JR    Z,noinv
        LD    B,16
invpig    LD    A,(HL)
        CPL
        LD    (HL),A
        DEC   HL
        LD    A,(HL)
        CPL
        LD    (HL),A
        DEC   HL
        DEC   HL
        DEC   HL
        DJNZ  invpig
noinv     POP   HL                ; xy of pigeon

```

```

POP    BC                                ; get xy of player
LD     A,L
SUB    C
LD     E,L
LD     D,A
JR     NC,abspos
NEG
abspos CP    6                            ; in visible range
JR     NC,noview                        ; offscreen
LD     A,D                            ; undo ABS
ADD    A,A
LD     H,A
ADD    A,14
LD     L,A
LD     A,H                            ; dy*2
NEG
LD     H,A
ADD    A,A                            ; dy*4
ADD    A,A                            ; dy*8
ADD    A,H                            ; dy*10
ADD    A,11+52                        ; based on middle+buffer
LD     H,#40
LD     (HL),A                        ; set pigeon visible
LD     (oldpigxy+1),HL                ; save this position
noview LD     A,58+11
SUB    B
SUB    B
LD     (#400F),A                      ; set car correct

LD     HL,frames
LD     A,(HL)
SUB    6
wfr    CP    (HL)
JR     NZ,wfr                        ; wait for display

LD     A,C
CP     103
JR     NZ,play                        ; level finished?
LD     A,(score+1)
LD     C,A
sc100  CALL addscore                    ; score to level*100
LD     A,(score+1)
CP     C
JR     Z,sc100
LD     A,(pglpcnt+1)
DEC    A
LD     C,A
DEC    C
JP     nextpig                        ; next level

play   PUSH BC                        ; ZXPAND
LD     BC,%11100000000000111
LD     A,#A0

```

```

OUT    (C),A
POP    DE
PUSH   DE
IN     A,(C)
LD     B,D
LD     C,E
ADD    A,A
JR     C,nozxup
INC    C
nozxup ADD    A,A
JR     C,nozxdown
DEC    C
nozxdown ADD    A,A
JR     C,nozxleft
INC    B
nozxleft ADD    A,A
JR     C,keyb
DEC    B

keyb    LD     HL,keycodes      ; point to keytable of control
        LD     DE,(lastk)      ; get full keycode
        CALL  dirtest          ; test direction
        JR     NZ,t2
        INC    C                ; up
t2       CALL  dirtest
        JR     NZ,t3
        DEC    C                ; down
t3       CALL  dirtest
        JR     NZ,t4
        INC    B                ; left
t4       CALL  dirtest
        JR     NZ,nomove
        DEC    B                ; right
nomove   CALL  field
        JR     NC,illmove      ; out of path
        POP    AF              ; drop old XY
        DEFB   62              ; hide retrieve old
illmove  POP    BC
progress LD    A,0
        CP     C
        JR     NC,pigcnt
        LD     A,C
        LD     (progress+1),A
        CALL  addscore

pigcnt   LD     A,0
        DEC    A
        LD     HL,pigeonxy+1
        JR     NZ,nopmove
        LD     A,(HL)
pglpcnt LD     A,10
        INC    (HL)
nopmove  LD     (pigcnt+1),A

```



```

LD    A, (HL)
CP    104
JP    Z, dead
JP    gameloop

dirtest LD    A, (HL)          ; fetch highbyte of keycode
        DEC    HL            ; point to lowbyte
        PUSH   HL            ; use HL for test, so save
        LD     L, (HL)       ; fetch lowbyte
        AND    A             ; reset carry
        LD     H, A
        SBC    HL, DE        ; test against pressed key
        POP    HL
        DEC    HL            ; point to next keycode
        RET              ; result of test is used else

        DEFW   0             ; right
        DEFW   0             ; left
        DEFW   0             ; down
        DEFB   0             ; up
keycodes DEFB   0             ; up

addscore LD    HL, score+4
ten      DEC    HL
        INC    (HL)
        LD     A, (HL)
        CP     38
        RET    NZ
        LD     (HL), 28
        JR     ten

sfield   CALL   field
        OR     D              ; set bit belonging to field
        LD     (HL), A
retcom   RET

rnd       LD     DE, 0         ; the seed
        LD     HL, (frames)   ; with some timing
        DEC    DE
        ADD    HL, DE
        LD     A, H
        AND    #1F
        LD     H, A
        LD     (rnd+1), HL    ; save seed
        LD     A, (HL)
fndrnd   SUB    3
        JR     NC, fndrnd
        ADD    A, 3
        RET

```

```

lowres    DEFB 118
score     DEFB 28,28,28,28,0
level     DEFB "U"-27,0      ; preset for UP key definition
hiscore    DEFB 28,28,28,28,118

```

```

field     LD    HL,data-1
          LD    A,B          ; B can be technically out of
          CP    5            ; fieldrange from UP/DOWN
          RET    NC          ; then return NOT USED
          PUSH  BC
          INC    B
          LD    A,C
          AND    7
          INC    A
          LD    E,A
          LD    A,C
          AND    #F8
          RRA
          RRA
          RRA                ; /8
          INC    A

```

```

ffield    INC    HL
          DEC    A
          JR     NZ,ffield
          LD    A,16
          DJNZ  ffield
          POP    BC
; HL points to right field, now find bit
          LD    D,1
          LD    A,(HL)

```

```

fldbit    RRC    D          ; shift bit for setting
          RLCA          ; shift field for testing
          DEC    E
          JR     NZ,fldbit
          LD    A,(HL)      ; fetch value for setting
          RET              ; C set when field is used

```

```

space     EQU    #4330-86-$
          DEFS   space

```

```

data      EQU    $
init      LD    IX,hr
          LD    HL,#4000
          LD    SP,#4400
          LD    DE,#C000
          LD    BC,1024
          LDIR                ; repair 48K bug
          LD    HL,udgidx
          LD    DE,#4004
          LD    C,24
          LDIR

```

```

        LD    B,110
        LD    HL,linedata
cls2    LD    (HL),C
        INC   HL
        DJNZ  cls2

        LD    HL,hrentry
        LD    (repairhr+1),HL
        JP    init2

; left pigeon, right car
; display in reversed order, start at bottom.

udgidx    DEFB 11+110,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns
          DEFB ns,ns

dirs      DEFB "D"-27,"L"-27,"R"-27,28

sp2       EQU 80-$(+data)      ; filler to set mazebuffer
          DEFS sp2

noshow    DEFW 0

;
;          car      pigeon
udgs      DEFB 255,255,3,128
          DEFB 255,191,5,192
          DEFB 225,191,15,128
          DEFB 194,223,39,200
          DEFB 200,95 ,115,156
          DEFB 198,239,59,184
          DEFB 225,239,251,190
          DEFB 128,03 ,63,248
          DEFB 128,01 ,127,252
          DEFB 128,01 ,31,240
          DEFB 140,25 ,63,248
          DEFB 146,37 ,15,224
          DEFB 173,91 ,7,192
          DEFB 173,219,3,128
          DEFB 243,231,2,128
          DEFB 255,255,0,0

```

```
basic      DEFB 0,1
           DEFW 0
           DEFB 249,212,28
           DEFB 126
           DEFB 143,0,18,0,0
dfile      EQU $
           DEFB 118,0

vars       DEFB 128

linedata   EQU #43E0-110-6
?
last       EQU $
```