TIC-80 Cheat Sheet

• DISPLAY	240x136 pixels, 16 color palette
• INPUT	4 gamepads with 8 btns/mouse/kbd
• SPRITES	256 8x8 tiles & 256 8x8 sprites
• MAP	240x136 cells, 1920x1088 pixels
• SOUND	waveforms x 4 channels

Memory Map, Key & Button Codes

ADDR	INFO	BYTES	COMMENTS
0	Screen	16320	240x136 x 4bit cl
3FC0	Palette	48	3(RGB) x 16 cls
3FF0	Palette Map	16	Palette map
3FF8	Border Color		
3FF9	Scrn Offset	2	X, Y offset
3FFB	Mouse Sprite	1	
3FFC	BLIT Segment	1	
3FFD	(RESERVED)	3	
4000	Tiles	8192	256 8x8 4bit BGs
6000	Sprites	8192	256 8x8 4bit SPRs
8000	Map	32640	240x136 map chips
FF80	Gamepads	4	Btn state
FF84	Mouse	4	Mouse state
FF88	Keyboard	4	Kbd state
FF8C	SFX State	16	
FF9C	Sound Reg	72	
FFE4	Waveforms	256	16 waves
100E4	SFX	4224	
11164	Music Patns	11520	
13E64	Music Tracks	408	
13FFC	Sound St	4	
14000	Stereo Vol	4	
14004	Persist Mem	1024	Persistent RAM
14404	Sprite Flags	512	
14604	System Font	2048	256 8x8 1bit font
14E04	Btns Mapping	32	
14E24	(RESERVED)	12764	

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Q	R	S	T	U	V	W	X	Y	Z	0	1	2	3	4	5
2	6	7	8	9	-	=	(	)	\	;	'	`	,	.	/	SP
3	TAB	RET	BKSP	DEL	INS	PGUP	PGDN	HOME	END	UP	DOWN	LEFT	RGHT	CAPS	CTRL	SHFT

↑	↓	←	→	A	B	X	Y
0	1	2	3	4	5	6	7

Control

ESC	Go back
F1-F5	Code,Sprite,Tile,Music,SFX editors
C-F ↑↓68	Find, Find Next/Prev
C-G	Go to line
C-S	Save
C-Tab	Indent line
C-O	Code outline

Commands

```
help(*) [commands|api|keys|buttons|…]
[*] try “help commands” to see available help
menu → enable “dev” mode, std/vi/emacs
new [lua|js|python|…] → create new code
edit → go to editor
folder → open the current dir in Mac/PC
save|load filename → save/load the code
```

Callbacks

```
TIC() → called per frame (60fps)
OVR() → called ever frame, overlay layer
BDR(row) → called per scan line, e.g. change
palette per scan line
MENU(index) → menu handler
BOOT → startup function
```

Input

```
btn(id) → pressed
btnp(code,hold=-1,period=-1) → released &
pressed since last frame | period
key(code) → pressed
keyp(code,hold=-1,period=-1) → released &
pressed since last frame | period
mouse() → x, y, left, mid, right, scrlx, scrly
```

Drawing

```
clip(x,y,w,h) : set clipping region
cls(color=0) : clear screen
circ(x,y,radian,color) : fill circle
circb(x,y,radian,color) : draw circle frame
line(x0,y0,x1,y1,color) : draw line
pix(x,y,color) : draw dot
pix(x,y) → color
rect(x,y,w,h,color) : fill rect
rectb(x,y,w,h,color) : draw rect frame
tri(x1,y1,x2,y2,x3,y3,color) : fill triangle
ttri(x1,y1,x2,y2,x3,y3,u1,v1,u2,v2,u3,v3,textsr
c=0,chromakey=-1,z1=0,z2=3,z3=0) : texture tri
elli(x,y,a,b,color) : fill ellipse
ellib(x,y,a,b,color) : draw ellipse frame
```

Program/Interrupts

```
exit() : exit app
reset() : reset
time() → ticks (milliseconds)
tstamp() → epoch seconds since 1970/1/1
trace(msg,color=15) : debug print
```

Sprite/Map

```
fget(spid,flg) → true if the flag is set
fset(spid,flg,bool) : set sprite flg[0..8]
map(x=0,y=0,w=30,h=17,sx=0,sy=0,colorkey=-1,sca
le=1,remap=nil) :
# x,y,w,h: rect of map tiles to draw
# colorkey: opaque (-1) or color index
# scale: scaling drawn tiles?
# remap: func(tile,x,y)->tile,flip,rot
mget(x,y) → tileid
mset(x,y,tleid) : set tile
spr(id,x,y,colorkey=-1,scale=1,flip=0,rotate=0,
w=1,h=1) :
# flip: 0,1,2,3 -> no,horiz,vert,both
# rotate: 0,1,2,3 -> 0, 90, 180, 270
```

Text

```
font(text,x,y,chromakey,char_width,char_height,
fixed=false,sclae=1) → width. draw text w/ fg
sprites
print(text,x=0,y=0,color=15,fixed=false,scale=1
,smallfont=false) → width. draw text w/ default
font
```

Memory

```
memcpy(dst,src,sz) : memcpy
memset(dst,value,sz) : memset→
peek(addr,bits=8) → value
peek{1|2|4}(addr) → value. 1,2,4 ver of peek
pmem(index,value) → save in persistent memory
pmem(index) → value from persistent memory
poke(addr,value,bits=8) : set value
poke{1|2|4}(addr,value) : 1,2,4 ver of poke
sync(mask=0,bank=0,tocart=false) : [pro version
only] bank switch
vbank([bank]) → prev. switch vbank.
```

Sound

```
music(track=-1,frame=-1,row=-1,loop=true,sustai
n=false,tempo=-1,speed=01) : play music
sfx(id,note=-1,dur=-1,chnl=0,vol=15,spd=0) :
sfx
```

Tutorial

Tracing

1. run TIC-80
2. new python
3. edit
4. add BOOT as below

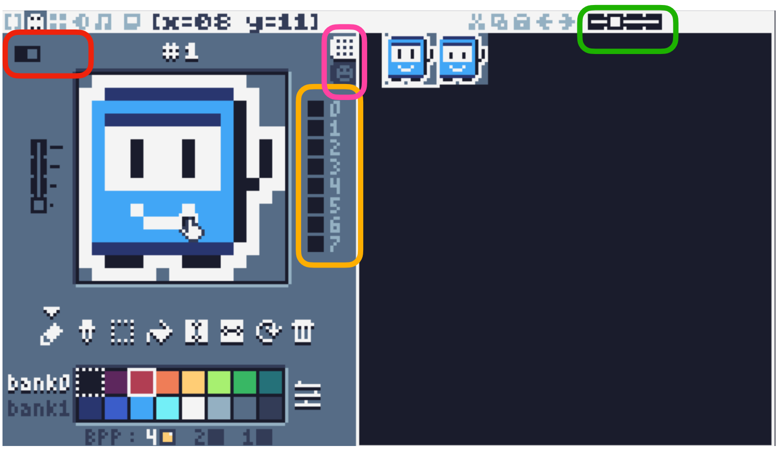
```
def BOOT():
 trace("BOOT!")
```

5. C-R to run
6. ESC to escape
7. Confirm "BOOT!" in console

Save & Load

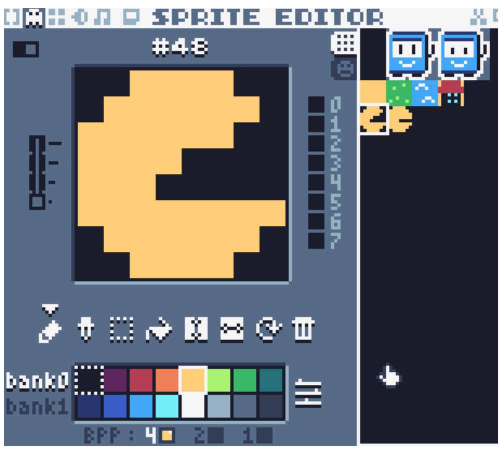
1. save foo # foo.tic saved
2. folder # finder/explorer opens the folder
3. load foo # foo.tic loaded

Sprite Editor



Advanced menu to show flags
Sprite size 1x1,2,2,3x3,4x4
Tiles or Sprites
Flags

Draw Sprites



To draw sprite #48, call spr().
colorkey=0 means the color 0(black) is treated as a transparent color.

```
def TIC():
 cls(13)
 spr(48,x,y,colorkey=0)
```

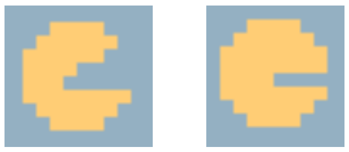
Sprite Animation

To draw sprite #48 and #49 animation,

set sid=48 or 49 and draw it by `spr` with colorkey=0 (black).

```
def TIC():
 cls(13)
  # set x, y…
 sid = 48 + t%60//30
 spr(sid,x,y,colorkey=0)
```

It'll show the sprite #48 and #49 alternately.



Hit Test

To test whether if 2 sprites a and b overlap, abs() is handy.

```
def hit(a, b):
 if abs(a.x-b.x)<width and abs(a.y-b.y)<height:
  return True
 else
  return False
```

Deleting a sprinte in an array

Deleting an item during iteration could cause a problem.

```
# global
missiles=[]
enemies=[]

def hit_test():
  # check hits and set missile or enemy's
"delme" property -> True if you want to delete
it

def delete_sprites():
 global missiles,enemies
 missiles=[x for x in missiles if
x.delme==False]
 enemies=[x for x in enemies if x.delme==False]

def TIC():
 …
 hit_test()
 delele_sprites()
```