TIC-80 workshop

Space Shooter

So Koide - Jan 26th, 2024 - Lightning Talks Special Event

AGENDA

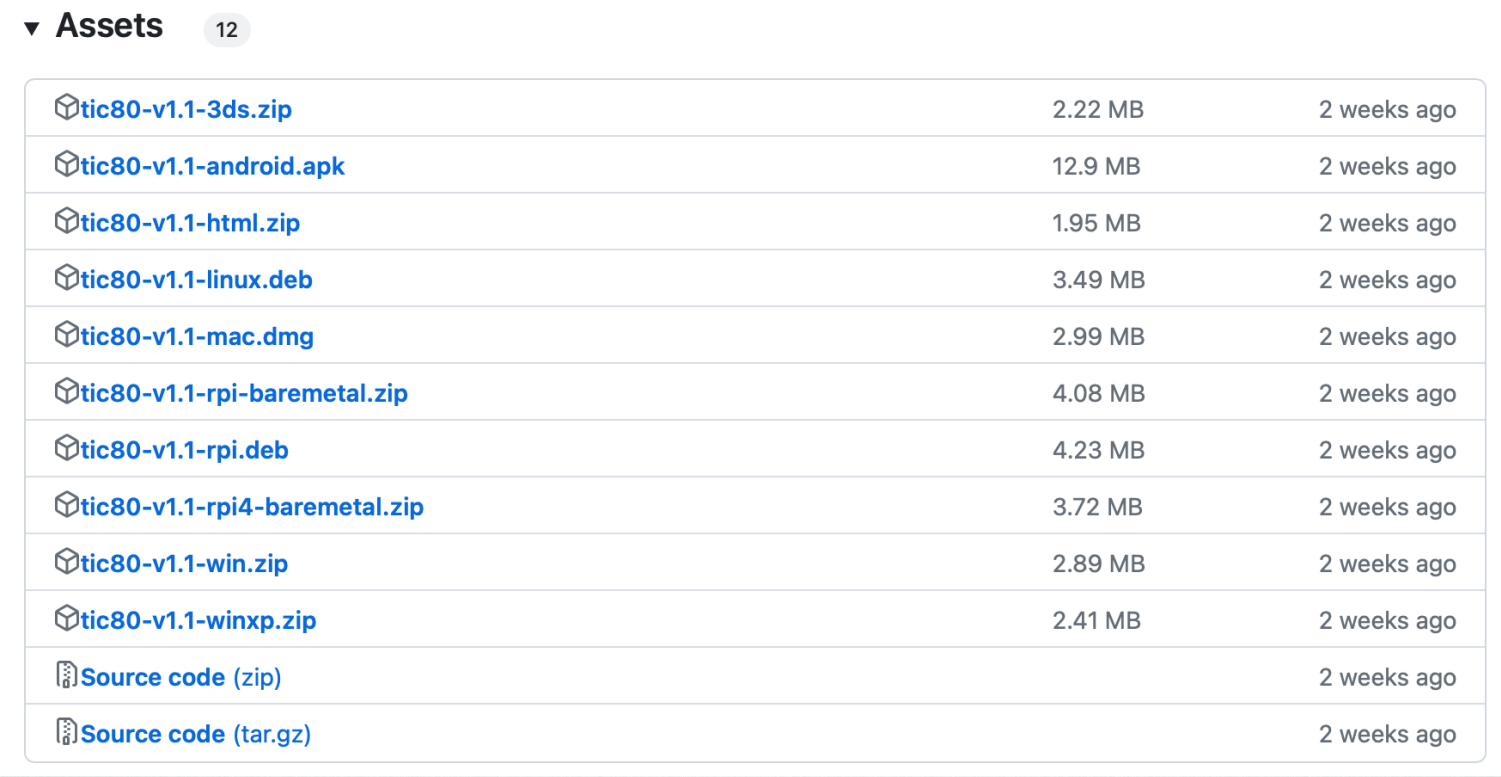
Everyone) Intro

1. Install TIC-80 at https://github.com/nesbox/TIC-80/releases
2. Download **TIC-80 cheat sheet.pdf** at https://github.com/sokoide/tic80-cheatsheet
3. Quick first game (help, folder, new, save load)

Optional) Step-by-step space shooter tutorial or make your own game

INSTALL

- https://github.com/nesbox/TIC-80 → Releases or build it

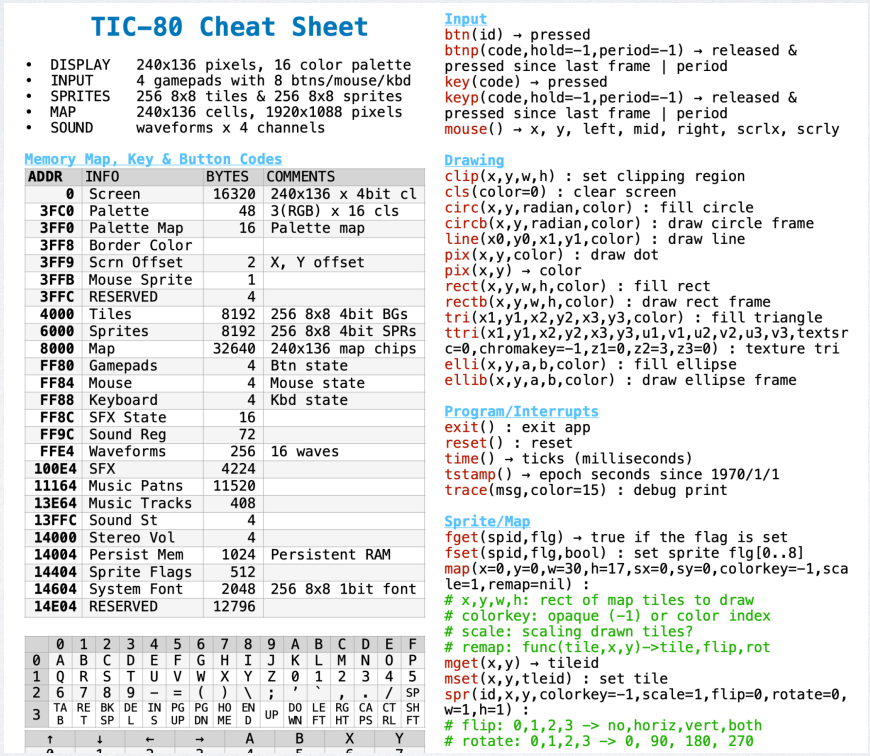


▼ **Assets** 12

tic80-v1.1-3ds.zip	2.22 MB	2 weeks ago
tic80-v1.1-android.apk	12.9 MB	2 weeks ago
tic80-v1.1-html.zip	1.95 MB	2 weeks ago
tic80-v1.1-linux.deb	3.49 MB	2 weeks ago
tic80-v1.1-mac.dmg	2.99 MB	2 weeks ago
tic80-v1.1-rpi-baremetal.zip	4.08 MB	2 weeks ago
tic80-v1.1-rpi.deb	4.23 MB	2 weeks ago
tic80-v1.1-rpi4-baremetal.zip	3.72 MB	2 weeks ago
tic80-v1.1-win.zip	2.89 MB	2 weeks ago
tic80-v1.1-winxp.zip	2.41 MB	2 weeks ago
Source code (zip)		2 weeks ago
Source code (tar.gz)		2 weeks ago

CHEAT SHEET

- https://github.com/sokoide/tic80-cheatsheet



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	RESERVED	4	
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	RESERVED	12796	

	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

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
```

QUICK FIRST GAME

help → help commands

```
>help

usage: help [<text>|version|welcome|
spec|ram|vram|commands|api|keys|buttons|
startup|terms|license]

type help commands to show commands

press ESC to switch editor/console

>help commands
Console commands:
cd cls config del demo dir edit eval
exit export folder help import load
menu mkdir new resume run save surf
>
```

CTRL|⌘ + R→ ESC

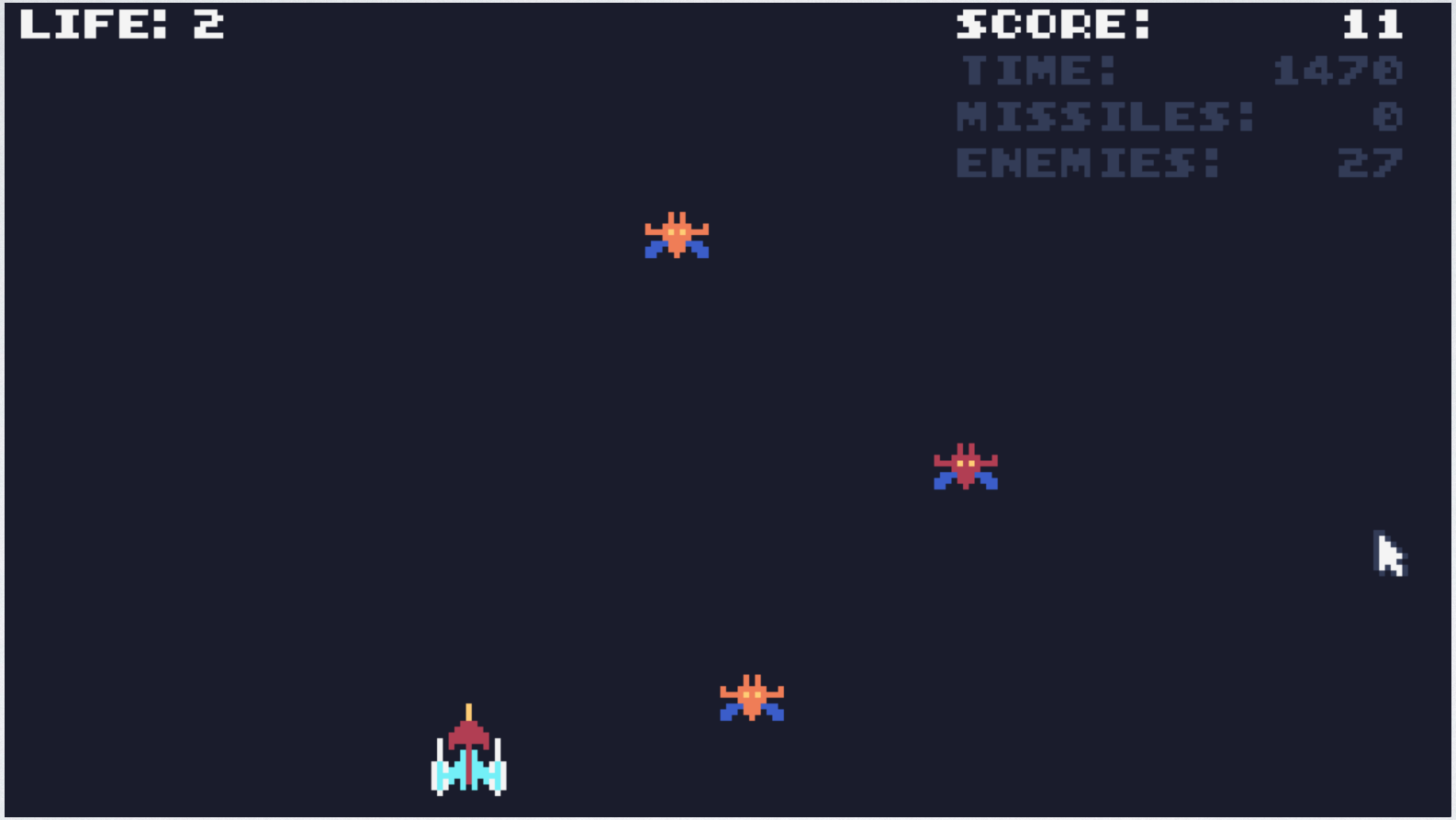
new python→ edit

```
>new python
new cart has been created
>edit
```

save, load, folder

```
>save first
cart first.tic saved!

>load first
cart first.tic loaded!
use RUN command to run it

>folder
Storage path:
/Users/scott/Library/Application
Support/com.nesbox.tic/TIC-80/
>
```

SPACE SHOOTER WORKSHOP

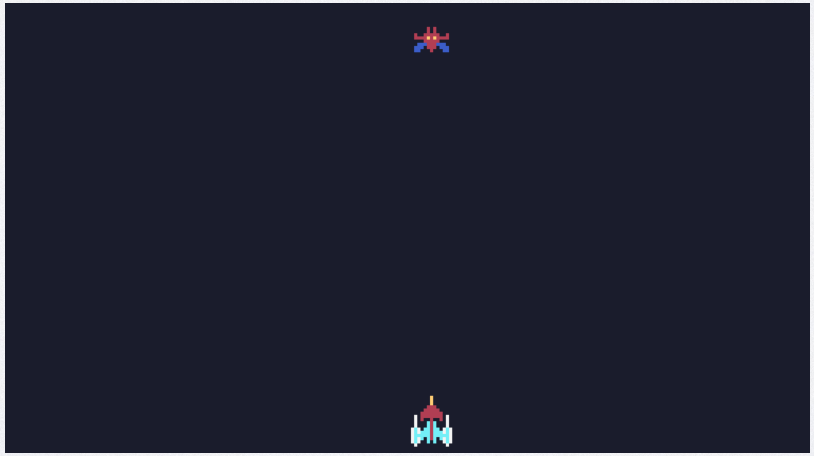


LOAD STARTER.TIC

- Download **starter.tic** at https://github.com/sokoide/tic80-cheatsheet
- load **starter.tic** → edit
- save **shooter**

DISPLAY YOUR SHIP

```
def BOOT():
 pass

def TIC():
 cls(0)
 # your ship
 spr(256,120,120,w=2,h=2,colorkey=0)
 # enemy
 spr(256+32,120,0,w=2,h=2,colorkey=0)
```



MOVE YOUR SHIP

```
class Sprite:
 def __init__(self,id,x,y):
  self.id=id
  self.x=x
  self.y=y

p=Sprite(256,120,120) #player
e=Sprite(256+32,120,0) #enemy
```

```
def handle_input():
 if btn(2):
  p.x-=1
 if btn(3):
  p.x+=1

def TIC():
 cls(0)
 handle_input()
 spr(p.id,p.x,p.y,w=2,h=2,colorkey=0)
 spr(e.id,e.x,e.y,w=2,h=2,colorkey=0)
```

FIRE MISSILES

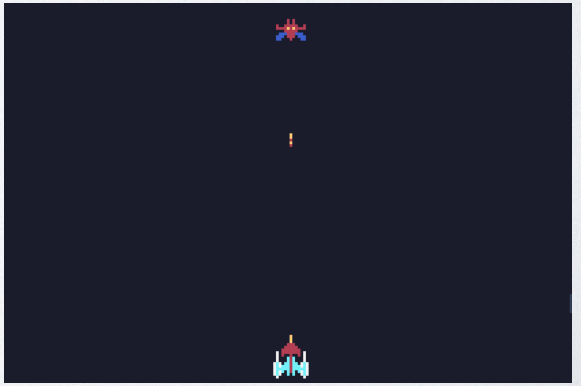
```
p=Sprite(256,120,120) #player
e=Sprite(256+32,120,0) #enemy
ms=[] # array of missiles
```

```
def fire():
 ms.append(Sprite(258,p.x,p.y-8))

def draw_missiles():
 for m in ms:
  spr(m.id,m.x,m.y,w=2,h=2,colorkey=0)
```

```
def move_missiles():
 for m in ms:
  m.y-=4
```

#TODO: we need to delete missiles if m.y<0
(next page)

```
def handle_input():
 if btn(2): # left
  p.x-=1
 if btn(3): # right
  p.x+=1
 if btnp(4): # Z key
  fire()

def TIC():
 cls(0)
 handle_input()
 move_missiles()
 draw_missiles()
```



DELETE MISSILES IF M.Y<0

```
class Sprite:
 def __init__(self,id,x,y):
  self.id=id
  self.x=x
  self.y=y
  self.delme=False
```

```
def TIC():
 cls(0)
 handle_input()
 move_missiles()
 delete_sprites()
 draw_missiles()
```

```
def move_missiles():
 for m in ms:
  m.y-=4
  if m.y<0:
   trace("Delete me "+str(m.y))
   m.delme=True
```

```
def delete_sprites():
 global ms
 ms=[x for x in ms if x.delme==False]
```

MORE ENEMIES

```
p=Sprite(256,120,120) #player
es=[Sprite(256+32,80,0),
 Sprite(256+32,120,0),
 Sprite(256+32,160,0)] #enemies
ms=[] # array of missiles
```

```
def TIC():
 cls(0)
 handle_input()
 move_missiles()
 delete_sprites()
 draw_missiles()
 spr(p.id,p.x,p.y,w=2,h=2,colorkey=0)
 for e in es:
  spr(e.id,e.x,e.y,w=2,h=2,colorkey=0)
```

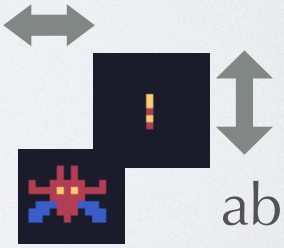
HIT TEST - YOUR MISSILE & ENEMIES

```
def TIC():
 cls(0)
 handle_input()
 move_missiles()
 hit_test()
 delete_sprites()
 draw_missiles()
```

```
def delete_sprites():
 global ms,es
 ms=[x for x in ms if x.delme==False]
 es=[x for x in es if x.delme==False]
```

```
def hit_test():
 for m in ms:
  for e in es:
   if hit(m,e):
    m.delme=True
    e.delme=True

def hit(a,b):
 if a.delme or b.delme:
  return False
 if abs(a.x-b.x)<16 and abs(a.y-b.y)<16
  return True
 return False
```

abs(a.x-b.x)



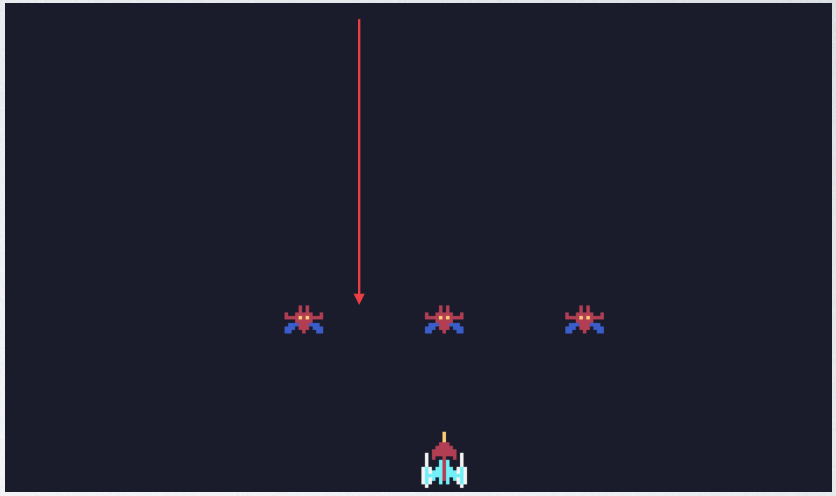
abs(a.y-b.y)

if overlapping,
both ABSs should be <16

MOVE ENEMIES

```
def TIC():
 cls(0)
 handle_input()
 move_missiles()
 move_enemies()
 hit_test()
```

```
def move_enemies():
 for e in es:
  e.y+=1
  if e.y<0:
   e.delme=True
```



HIT TEST - ENEMIES & YOUR SHIP

```
# global vars
p=Sprite(256,120,120) #player
es=[Sprite(256+32,80,0),
 Sprite(256+32,120,0),
 Sprite(256+32,160,0)] #enemies
ms=[] # array of missiles
gameover=-1
```

```
def hit_test():
 global gameover
 for m in ms: # missiles & enemies
  for e in es:
   if hit(m,e):
    m.delme=True
    e.delme=True
 for e in es: # enemies & your ship
  if hit(e,p):
   e.delme=True
   gameover=180
```

CONT'D: GAMEOVER

```
def TIC():
 global gameover
 cls(0)
 if gameover>=0:
  print("GAME OVER",x=90,y=60,color=12)
  gameover-=1
  if gameover==0:
   reset()
 else:
  handle_input()
  move_missiles()
  move_enemies()
  hit_test()
  delete_sprites()
  draw_missiles()
  spr(p.id,p.x,p.y,w=2,h=2,colorkey=0)
  for e in es:
```

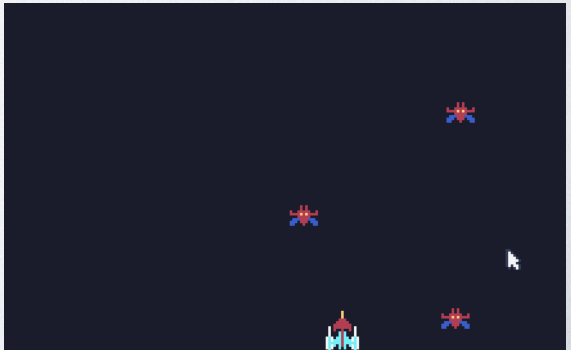
SPAWN ENEMIES

```
# global vars
p=Sprite(256,120
es=[Sprite(256+3
 Sprite(256+32,1
 Sprite(256+32,1
ms=[] # array of
gameover=-1
t=0
```

```
def TIC():
 global gameover,t
 t+=1
 cls(0)
 if gameover>=0:
  print("GAME OVER"
  gameover-=1
  if gameover==0:
   reset()
 else:
  spawn_enemies()
  handle_input()
```

```
def spawn_enemies():
 if t%40==0:
  x=random.randint(0,240-16)
  es.append(Sprite(256+32,x,0))
```

```
# title:   space sh
# author:  sokoide
# desc:    space sh
# site:    https://
# license: MIT Lice
# version: 0.1
# script:  python
import random
```



MORE FEATURES!

- Lives
- Scoring
- Different enemy character
- Enemy animation
- Explosion animation
- Internal information on top-right
- …
- **final.tic** is available at https://github.com/sokoide/tic80-cheatsheet for your reference