

Grid obj. has: (whole game)  
- side length (in tiles) inc  
- tile size (in px) inc  
- score inc  
- load\_images()

Snake obj. has:  
- parts x array [ ] inc [side.len]<sup>2</sup>  
- parts y array [ ] inc [ ]  
- head\_dir 2 ← → 0 inc  
- snake\_length 1 inc  
- head\_x inc  
- head\_y inc

Apple obj.:

- apple\_x inc  
- apple\_y inc  
- regen\_apple()  
{ random (0 → side.len) = app\_x  
" " " " = app\_y }

0 10px 20 30 40 50 60 70 (x tile size = 10)

0									
1									
2									
3									
4									
5									
6									
7									

e.g. apple at (3, 7)  
⇒ 30px in x,  
70px in y

game loop

update objects: apple, snake parts,

move head → ck inside board  
ck if pos = any snake part  
true → game over()  
false → ck if apple x,y = head new x,y  
true → regen\_apple(), add 5 to score, add 1 to sk\_len  
false → cont.

update\_snake\_parts()  
for i in sk\_len:  
sk\_x[i] = sk\_x[i-1] (go backwards)  
sk\_y[i] = sk\_y[i-1]

render();

take x, y for entities and  
x tile\_size for pixel positions.

arr[2, 4, 3, 5]  
⇒ arr[3] = arr[3-1]  
∴ arr = [2, 4, 3, 3]  
⇒ [2, 4, 4, 3]  
⇒ [2, 2, 4, 3]  
now update arr[0] with new head x, y

priv. check\_collisions() → board → snake → apple  
may game over()

priv. load\_images()  
priv. init\_game();  
priv. more();

@Override paintComponent()  
render()

@Override actionPerformed()  
more() → collisions() → repaint()

TAdaptor() extends KeyAdaptor  
@Override keyPressed(KeyEvent k)