

## py

```
1 import tkinter
2 from math import floor
3
4 class TicTacToe():
5     def __init__(self, grid_size=100, width=5, height=4, addr_to='127.0.0.1'):
6         self.grid_size = grid_size
7         self.width = width
8         self.height = height
9         self.addr_to = (addr_to, 5005)
10
11         self.root = tkinter.Tk()
12         self.root.resizable(False, False)
13         self.canvas = tkinter.Canvas(self.root, width=grid_size*width, height=grid_size*height)
14         self.canvas.pack()
15
16         self.root.mainloop()
17
18 if __name__ == '__main__':
19     TicTacToe()
```

## draw\_grid

```
14         self.canvas.pack()
15
16 +         self.draw_grid()
17 +
18         self.root.mainloop()
19 +
20 +     def draw_grid(self):
21 +         s = self.grid_size
22 +         for y in range(self.height):
23 +             for x in range(self.width):
24 +                 self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
25 +                 self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
26 +         self.canvas.update()
27
28 if __name__ == '__main__':
```

## recv

```
1 import tkinter
2 from math import floor
3 +import socket
4 +import threading
5
6 class TicTacToe():
```

```
16         self.canvas.pack()
17
18 +         self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
19 +         self.sock.bind(('0.0.0.0', 5005))
20 +
21 +         thread = threading.Thread(target=self.recv)
22 +         thread.daemon=True
23 +         thread.start()
24 +
25         self.draw_grid()
26
```

```
35         self.canvas.update()
36
37 +     def recv(self):
38 +         while True:
39 +             data, self.addr_to = self.sock.recvfrom(1024)
40 +             print(f"received bytes: {data} from {self.addr_to}")
41 +
42 if __name__ == '__main__':
43     TicTacToe()
```

## recv\_draw

```
39         data, self.addr_to = self.sock.recvfrom(1024)
40         print(f"received bytes: {data} from {self.addr_to}")
41 +         data = data.decode('utf8').strip().split(',')
42 +
43 +         x, y, fill = int(data[0]), int(data[1]), data[2]
44 +         s = self.grid_size
45 +         self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
46
47 if __name__ == '__main__':
```

## mouse

```
23         thread.start()
24
25 +         self.root.bind('<ButtonPress>', self.mouse_click)
26 +
27 +         self.fill = 'black'
28 +
29         self.draw_grid()
30
31         self.root.mainloop()
32 +
33 +     def mouse_click(self, event):
34 +         x, y = floor(event.x/self.grid_size), floor(event.y/self.grid_size)
35 +         data = ','.join(map(str, (x,y,self.fill))).encode('utf8')
36 +         self.sock.sendto(data, self.addr_to)
37
38     def draw_grid(self):
```

## local\_state

```
27         self.fill = 'black'
28
29 +         self.grid = ['' for i in range(self.width*self.height)]
30         self.draw_grid()
31
```

```
34     def mouse_click(self, event):
35         x, y = floor(event.x/self.grid_size), floor(event.y/self.grid_size)
36 -         data = ','.join(map(str, (x,y,self.fill))).encode('utf8')
37 +         self.grid[x+y*self.width] = self.fill
38 +         data = ','.join(self.grid).encode('utf8')
39 +         self.sock.sendto(data, self.addr_to)
40 +         self.draw_grid()
41
42 +     def draw_grid(self):
43 +         self.canvas.delete(tkinter.ALL)
44 +         s = self.grid_size
45 +         for y in range(self.height):
46 +             for x in range(self.width):
47 +                 fill = self.grid[x+y*self.width]
48 +                 if fill:
49 +                     self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
50 +                 self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
51 +                 self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
```

```
57         data = data.decode('utf8').strip().split(',')
58
59 -         x, y, fill = int(data[0]), int(data[1]), data[2]
60 -         s = self.grid_size
61 -         self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
62 +         self.grid[:] = data
63 +         self.draw_grid()
64
65 if __name__ == '__main__':
```

## 2players

```

17     self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
18     self.sock.bind(('0.0.0.0', 5005))
19 +     try:
20 +         self.sock.bind(('0.0.0.0', 5005))
21 +     except OSError as ex:
22 +         print('cant bind socket - is another client is already running on this port?')
23 +         self.sock.sendto(b'', self.addr_to)
24
25     thread = threading.Thread(target=self.recv)


29     self.root.bind('<ButtonPress>', self.mouse_click)
30
31     self.fill = 'black'
32 +     self.fill = '0'
33
34     self.grid = [[' ' for i in range(self.width*self.height)]]


38     def mouse_click(self, event):
39         x, y = floor(event.x/self.grid_size), floor(event.y/self.grid_size)
40 +         if (not any(self.grid)):
41 +             self.fill = 'X'
42         self.grid[x+y*self.width] = self.fill
43         data = ','.join(self.grid).encode('utf8')


51         for x in range(self.width):
52             fill = self.grid[x+y*self.width]
53             if fill:
54 +             if fill == 'X':
55 +                 self.canvas.create_line(x*s, y*s, (x+1)*s, (y+1)*s, fill="red", width=4)
56 +                 self.canvas.create_line(x*s, (y+1)*s, (x+1)*s, y*s, fill="red", width=4)
57 +             elif fill == '0':
58 +                 self.canvas.create_oval(x*s, y*s, (x+1)*s, (y+1)*s, outline='blue', width=4)
59 +             else:
60                 self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
61                 self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
62                 self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
63             self.canvas.update()


66         print(f"received bytes: {data} from {self.addr_to}")
67         data = data.decode('utf8').strip().split(',')
68 +         if not any(data):
69 +             continue
70
71         self.grid[:] = data

```

## commandline

```

73
74 if __name__ == '__main__':
75     TicTacToe()
76 +     import sys
77 +     if len(sys.argv) == 1:
78 +         sys.argv.append('127.0.0.1')
79 +     TicTacToe(addr_to=sys.argv[1])

```