py

```
import tkinter
from math import floor

class TicTacToe():
    def __init__(self, grid_size=100, width=5, height=4, addr_to='127.0.0.1'):
        self.grid_size = grid_size
        self.width = width
        self.height = height
        self.addr_to = (addr_to, 5005)

        self.root = tkinter.Tk()
        self.root.resizable(False, False)
        self.canvas = tkinter.Canvas(self.root, width=grid_size*width, height=grid_size*height)
        self.canvas.pack()

        self.root.mainloop()

if __name__ == '__main__':
    TicTacToe()
```

draw_grid

```
        self.canvas.pack()

+       self.draw_grid()
+
        self.root.mainloop()
+
+   def draw_grid(self):
+       s = self.grid_size
+       for y in range(self.height):
+           for x in range(self.width):
+               self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
+               self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
+       self.canvas.update()

if __name__ == '__main__':
```

recv

```
import tkinter
from math import floor
+import socket
+import threading

class TicTacToe():
```

```
        self.canvas.pack()

+       self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
+       self.sock.bind(('0.0.0.0', 5005))
+
+       thread = threading.Thread(target=self.recv)
+       thread.daemon=True
+       thread.start()
+
        self.draw_grid()

```

```
        self.canvas.update()

+   def recv(self):
+       while True:
+           data, self.addr_to = self.sock.recvfrom(1024)
+           print(f"received bytes: {data} from {self.addr_to}")
+
if __name__ == '__main__':
    TicTacToe()
```

recv_draw

```
            data, self.addr_to = self.sock.recvfrom(1024)
            print(f"received bytes: {data} from {self.addr_to}")
+           data = data.decode('utf8').strip().split(',')
+
+           x, y, fill = int(data[0]), int(data[1]), data[2]
+           s = self.grid_size
+           self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)

if __name__ == '__main__':
```

mouse

```
        thread.start()

+       self.root.bind('<ButtonPress>', self.mouse_click)
+
+       self.fill = 'black'
+
        self.draw_grid()

        self.root.mainloop()
+
+   def mouse_click(self, event):
+       x, y = floor(event.x/self.grid_size), floor(event.y/self.grid_size)
+       data = ','.join(map(str,(x,y,self.fill))).encode('utf8')
+       self.sock.sendto(data, self.addr_to)

    def draw_grid(self):
```

local_state

```
        self.fill = 'black'

+       self.grid = ['' for i in range(self.width*self.height)]
        self.draw_grid()

```

```
    def mouse_click(self, event):
        x, y = floor(event.x/self.grid_size), floor(event.y/self.grid_size)
-       data = ','.join(map(str,(x,y,self.fill))).encode('utf8')
+       self.grid[x+y*self.width] = self.fill
+       data = ','.join(self.grid).encode('utf8')
        self.sock.sendto(data, self.addr_to)
+       self.draw_grid()

    def draw_grid(self):
+       self.canvas.delete(tkinter.ALL)
        s = self.grid_size
        for y in range(self.height):
            for x in range(self.width):
+               fill = self.grid[x+y*self.width]
+               if fill:
+                   self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
                self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
                self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
```

```
            data = data.decode('utf8').strip().split(',')

-           x, y, fill = int(data[0]), int(data[1]), data[2]
-           s = self.grid_size
-           self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
+           self.grid[:] = data
+           self.draw_grid()

if __name__ == '__main__':
```

2players

```
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
-       self.sock.bind(('0.0.0.0', 5005))
+       try:
+           self.sock.bind(('0.0.0.0', 5005))
+       except OSError as ex:
+           print('cant bind socket - is another client is already running on this port?')
+           self.sock.sendto(b'', self.addr_to)

        thread = threading.Thread(target=self.recv)
```

```
        self.root.bind('<ButtonPress>', self.mouse_click)

-       self.fill = 'black'
+       self.fill = 'O'

        self.grid = ['' for i in range(self.width*self.height)]
```

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

```
            for x in range(self.width):
                fill = self.grid[x+y*self.width]
-               if fill:
+               if fill == 'X':
+                   self.canvas.create_line(x*s, y*s, (x+1)*s, (y+1)*s, fill="red", width=4)
+                   self.canvas.create_line(x*s, (y+1)*s, (x+1)*s, y*s, fill="red", width=4)
+               elif fill == 'O':
+                   self.canvas.create_oval(x*s, y*s, (x+1)*s, (y+1)*s, outline='blue', width=4)
+               else:
                    self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, fill=fill)
                self.canvas.create_rectangle(x*s, y*s, (x+1)*s, (y+1)*s, outline='black', width=4)
-               self.canvas.create_text(x*s+s/2, y*s+s/2, text=f'{x},{y}', fill='black')
        self.canvas.update()

```

```
            print(f"received bytes: {data} from {self.addr_to}")
            data = data.decode('utf8').strip().split(',')
+           if not any(data):
+               continue

            self.grid[:] = data
```

commandline

```

 if __name__ == '__main__':
-    TicTacToe()
+    import sys
+    if len(sys.argv) == 1:
+        sys.argv.append('127.0.0.1')
+    TicTacToe(addr_to=sys.argv[1])
```