```python
import json
import socket
import threading
import tkinter as tk


class TcpClient(threading.Thread):
    def __init__(self, gui, ip, port, byte_num):
        super().__init__()
        self.__gui = gui
        self.k_sock = None
        self.__ip = ip
        self.__port = port
        self.__bytes = byte_num

    def run(self):
        self.k_sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        self.k_sock.connect((self.__ip, self.__port))
        try:
            while True:
                data = self.k_sock.recv(self.__bytes)
                data_dict = json.loads(data.decode())
                print("Client data: {}".format(data_dict))
                self.__gui.display_led_state(data_dict["led1"], data_dict["led2"])

        finally:
            self.k_sock.close()


class ClientGui(tk.Tk):
    def __init__(self, ip, port, byte_num):
        super().__init__()
        # Übergabe des TCP Clients
        self.__tcp_client = TcpClient(self, ip, port, byte_num)
        self.__tcp_client.start()
        # Titel und Maße des Fensters
        self.title("Lichtsteuerung")
        self.geometry("550x80")
        # Frame für Raum 1 wird generiert
        self.lbl = tk.LabelFrame(self, text="Raum 1", width=250, height=80)
        self.lbl.grid(row=0, column=0, columnspan=2, sticky="W", padx=5, pady=0, ipadx=0, ipady=0)
        # Einschalt-Button für Raum 1
        self.an_btn = tk.Button(self.lbl, text="Einschalten", command=self.cb_an_r1)
        self.an_btn.grid(column=0, row=1, padx=15, pady=10)
        # Ausschalt-Button für Raum 1
```

```
46          self.aus_btn = tk.Button(self.lbl, text="Ausschalten",
    command=self.cb_aus_r1)
47          self.aus_btn.grid(column=1, row=1, padx=15, pady=10)
48          # LED-Anzeige für Raum 1
49          self.anzeige1 = tk.Button(self.lbl, text="     ", bg="
    grey", command="")
50          self.anzeige1.grid(column=2, row=1, padx=15, pady=10)
51          # Frame für Raum 2 wird generiert
52          self.lbl2 = tk.LabelFrame(self, text="Raum 2", width=
    250, height=80)
53          self.lbl2.grid(row=0, column=2, columnspan=2, sticky="W
    ", padx=5, pady=0, ipadx=0, ipady=0)
54          # Einschalt-Button für Raum 2
55          self.an_btn = tk.Button(self.lbl2, text="Einschalten",
    command=self.cb_an_r2)
56          self.an_btn.grid(column=0, row=1, padx=15, pady=10)
57          # Ausschalt-Button für Raum 2
58          self.aus_btn = tk.Button(self.lbl2, text="Ausschalten"
    , command=self.cb_aus_r2)
59          self.aus_btn.grid(column=1, row=1, padx=15, pady=10)
60          # LED-Anzeige für Raum 2
61          self.anzeige2 = tk.Button(self.lbl2, text="     ", bg="
    grey", command="")
62          self.anzeige2.grid(column=2, row=1, padx=15, pady=10)
63          self.raeume = {"R1": 0, "R2": 0}
64          self.__tcp_client.k_sock.send('{"init": 1}'.encode())
65
66
67      def __send(self):
68          msg = json.dumps(self.raeume)
69          self.__tcp_client.k_sock.send(msg.encode())
70
71      def display_led_state(self, led1_state, led2_state):
72          """
73          Set displayed LED states
74          :param led1_state: State of LED1
75          :param led2_state: State of LED2
76          """
77          try:
78              if led1_state and not led2_state:
79                  print("Client macht LED1 gelb und LED2 grau")
80                  self.anzeige1.configure(bg="yellow")
81                  self.anzeige2.configure(bg="grey")
82                  self.raeume["R1"] = 1
83                  self.raeume["R2"] = 0
84              elif not led1_state and led2_state:
85                  print("Client macht LED2 gelb und LED1 grau")
86                  self.anzeige1.configure(bg="grey")
87                  self.anzeige2.configure(bg="yellow")
88                  self.raeume["R1"] = 0
```

```
89                  self.raeume["R2"] = 1
90              elif led1_state and led2_state:
91                  print("Client macht LED2 gelb und LED1 gelb")
92                  self.anzeige1.configure(bg="yellow")
93                  self.anzeige2.configure(bg="yellow")
94                  self.raeume["R1"] = 1
95                  self.raeume["R2"] = 1
96              else:
97                  print("Client macht LED2 grau und LED1 grau")
98                  self.anzeige1.configure(bg="grey")
99                  self.anzeige2.configure(bg="grey")
100                 self.raeume["R1"] = 0
101                 self.raeume["R2"] = 0
102         except Exception as e:
103             print(e)
104
105     # Callback_Funktion um LED für Raum 1 einzuschalten
106     def cb_an_r1(self):
107         self.raeume["R1"] = 1
108         self.__send()
109
110     # Callback_Funktion um LED für Raum 1 auszuschalten
111     def cb_aus_r1(self):
112         self.raeume["R1"] = 0
113         self.__send()
114
115     # Callback_Funktion um LED für Raum 2 einzuschalten
116     def cb_an_r2(self):
117         self.raeume["R2"] = 1
118         self.__send()
119
120     # Callback_Funktion um LED für Raum 2 auszuschalten
121     def cb_aus_r2(self):
122         self.raeume["R2"] = 0
123         self.__send()
124
```