Założenia aplikacji wirtualnego panelu do wizualizacji i wprowadzania danych cyfrowych

24.07.2024

Celem aplikacji „panel” są: wizualizacja sygnałów cyfrowych symulatora oraz wprowadzanie sygnałów cyfrowych do symulatora. Sygnał cyfrowy może przyjmować jedną z wartości 0 i 1.

Założenia

- Położenie elementów wyświetlanych w panelu jest wczytywane z pliku konfiguracyjnego po uruchomieniu aplikacji. Nazwa pliku konfiguracyjnego jest przekazywana jako parametr wywołania programu panel.
- Informacja o stanie sygnałów cyfrowych jest przekazywana drogą sieciową, zwykle w obrębie jednego komputera.

Elementy panelu

Elementy jakie mogą wystąpić (być wyświetlane) w panelu to:

1. Etykieta - TEXT
2. Dioda LED - LED
3. Linijka LED - BAR
4. Przełącznik dwustanowy - SWITCH
5. Przycisk monostabilny - BUTTON

Każdy element wyświetlany w panelu musi posiadać parametry:

- dev - Identyfikator typu elementu: TEXT, LED, BAR, SWITCH, BUTTON.
- x, y - Położenie elementu w panelu w postaci współrzędnych x,y (0,0 w lewym górnym oknie panelu).

Elementy w zależności od typu mogą posiadać dodatkowe parametry:

TEXT

- text – treść wyświetlanego tekstu
- size – rozmiar czcionki
- color – kolor tekstu

LED

- size – rozmiar diody – średnica okręgu
- color – kolor diody (red, green, yellow, blue, white)

BAR

- len – liczba diod
- size – szerokość diody (proporcje wysokość : szerokość = 3:1
- color – kolor diody (red, green, yellow, blue, white)

SWITCH

- size – szerokość prostokąta

BUTTON

- active – stan generowany po naciśnięciu (0 lub 1)
- size – bok kwadratu

Dodatkowo w pliku konfiguracyjnym musi wystąpić „element” GLOBAL opisujący rozmiar okna panelu oraz tytuł wyświetlany w górnym pasku aplikacji. Element GLOBAL nie posiada współrzędnych x,y.

GLOBAL

- title – nazwa wyświetlana na górnym pasku aplikacji
- geometry – rozmiary i położenie okna

W pliku konfiguracyjnym mogą występować komentarze analogiczne do komentarzy w języku Python.

Zachowanie elementów panelu.

TEXT

Element statyczny służący do opisu elementów aktywnych. Parametry size i color powinny być opcjonalne. Jako wartości domyślne należy przyjąć kolor czarny i wielkość pasującą do rozmiaru innych wyświetlanych elementów.

LED

Dioda powinna odpowiadać okrągłej diodzie LED o średnicy 3 mm. Wyświetlany kolor powinien odpowiadać stanowi logicznemu w jakim znajduje się dioda. W przypadku diody czerwonej w stanie 1 należy wyświetlać koło o kolorze jasno czerwonym, a w przypadku diody w stanie 0 kolor (bardzo) ciemno czerwony, analogicznie dla pozostałych kolorów.

BAR

Element BAR symuluje pasek złożony z prostokątnych diod LED o wymiarach 2x5 mm. Liczba diod określona jest przez parametr size. Zasada wyświetlana diod analogiczna jak w przypadku diody okrągłej.

SWITCH

Element jest odpowiednikiem tzw. wyłącznika hebelkowego, w którym można zmieniać stan pomiędzy 0 i 1. Jego wygląd na panelu musi wskazywać w jakim aktualnie stanie się znajduje ON/OFF czyli 1/0

BUTTON

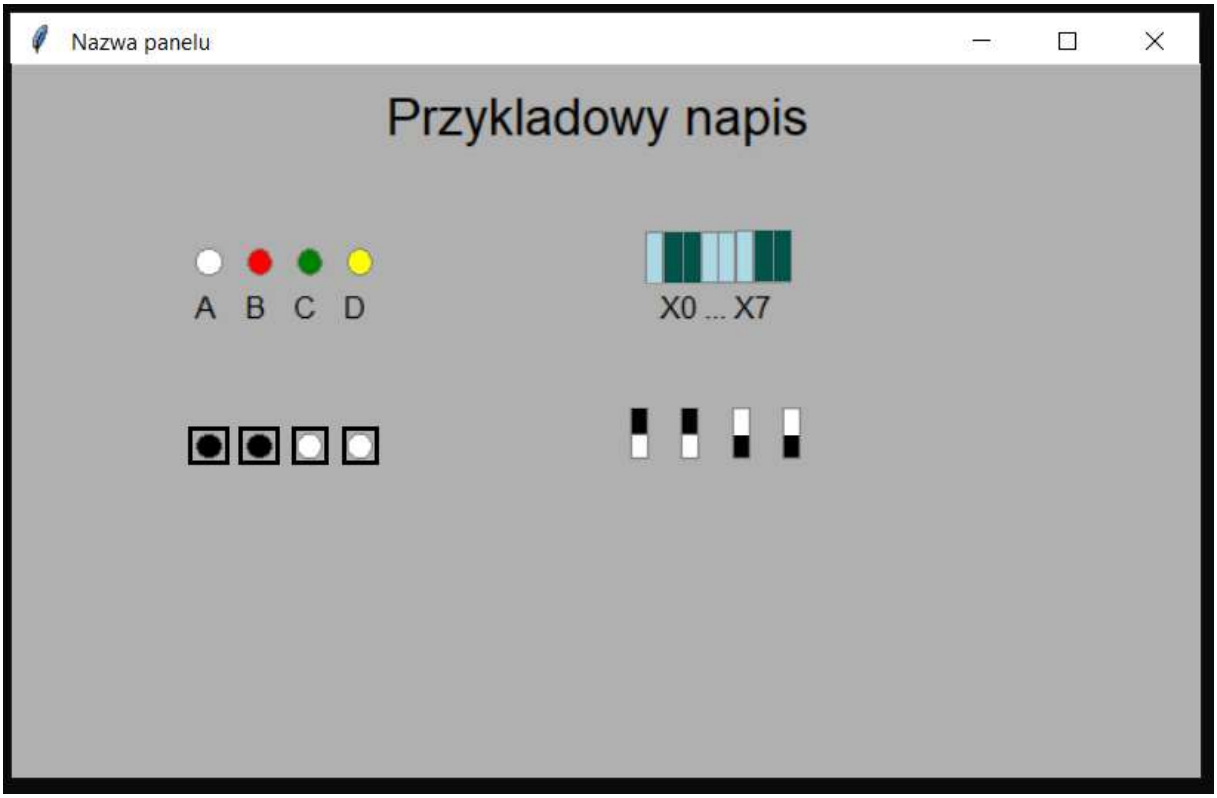
Element ten odpowiada microswitch-owi, którego naciśnięcia zmienia jego stan na aktywny, a po zwolnieniu przycisku stan wraca do poprzedniego. W panelu mogą wystąpić dwa rodzaje takich elementów różniące się stanem aktywnym: generujące stan 0 i generujące stan 1. Wygląd przycisku musi wskazywać jaki jest stan aktywny.

Przykład pliku konfiguracyjnego

```
{ 'dev': 'GLOBAL', 'title': 'Nazwa panelu', 'geometry': '642x385+200+400' }

{ 'dev': 'TEXT', 'text': 'Przykladowy napis', 'size': 24, 'color': 'black', 'x': 350, 'y': 30 }

{ 'dev': 'LED', 'size': 15, 'x': 110, 'y': 110, 'color': 'white' }
{ 'dev': 'LED', 'size': 15, 'x': 140, 'y': 110, 'color': 'red'   }
{ 'dev': 'LED', 'size': 15, 'x': 170, 'y': 110, 'color': 'green' }
```

```
{ 'dev': 'LED', 'size': 15, 'x': 200, 'y': 110, 'color': 'yellow'}

{ 'dev': 'TEXT', 'text': 'A', 'size': 14, 'color': 'black', 'x': 115, 'y': 145 }
{ 'dev': 'TEXT', 'text': 'B', 'size': 14, 'color': 'black', 'x': 145, 'y': 145 }
{ 'dev': 'TEXT', 'text': 'C', 'size': 14, 'color': 'black', 'x': 175, 'y': 145 }
{ 'dev': 'TEXT', 'text': 'D', 'size': 14, 'color': 'black', 'x': 205, 'y': 145 }
{ 'dev': 'TEXT', 'text': 'X0 ... X7', 'size': 14, 'color': 'black', 'x': 420, 'y': 145 }

{ 'dev': 'BAR', 'len': 8, 'size': 15, 'x': 378, 'y': 100, 'color': 'blue' }

{ 'dev': 'BUTTON', 'size': 15, 'x': 110, 'y': 220, 'active': 1 }
{ 'dev': 'BUTTON', 'size': 15, 'x': 140, 'y': 220, 'active': 1 }
{ 'dev': 'BUTTON', 'size': 15, 'x': 170, 'y': 220, 'active': 0 }
{ 'dev': 'BUTTON', 'size': 15, 'x': 200, 'y': 220, 'active': 0 }

{ 'dev': 'SWITCH', 'size': 30, 'x': 370, 'y': 205 }
{ 'dev': 'SWITCH', 'size': 30, 'x': 400, 'y': 205 }
{ 'dev': 'SWITCH', 'size': 30, 'x': 430, 'y': 205 }
{ 'dev': 'SWITCH', 'size': 30, 'x': 460, 'y': 205 }
```

Odpowiadający temu plikowi panel wygląda następująco:



W panelu znajdują się (poza napisami) 4 diody LED w stanie 1, 8 elementowy bar, 4 button-y (dwa generujące stan 1 i dwa generujące stan 0), cztery switch-e (dwa w stanie 1 i dwa w stanie 0)

Sterowanie diodami i bar-ami

Sterowanie diodami oraz bar-ami odbywa się poprzez przesłanie pakietów UDP na porcie 9001.

Przykład programu wysyłającego losowy stan elementów z przykładowego panelu (przydatny podczas testów) wygląda następująco:

```
from time import sleep
from random import randint
import socket
from socket import AF_INET, SOCK_DGRAM

server_socket = socket.socket(AF_INET, SOCK_DGRAM)
N = 12 # liczba wyswietlanych zmiennych

for _ in range(10):
  x = bytearray([randint(0,1) for n in range(N)])
  print('==',x)
  server_socket.sendto(x, ("localhost", 9001))
  sleep(5)
```

Po stronie panelu kod odbierający przychodzące dane na przykład wygląda tak:

```
import socket
from socket import AF_INET, SOCK_DGRAM

server_socket = socket.socket(AF_INET, SOCK_DGRAM)
server_socket.bind(("localhost", 9001))

while True:
  buffer, address = server_socket.recvfrom(256)
  print(list(buffer))
```

Kolejne bajty odpowiadają kolejnym elementom typu LED lub BAR w pliku konfiguracyjnym.

Wysyłanie informacji o stanie buton-ów i switch-y

Analogicznie wygląda sposób przekazywania informacji o naciśniętych buton-ach czy też stanie w jakim znajdują się switch-e. Każda zmiana dowolnego buton-a lub switch-a wysyła pakiet UDP zawierający tablicę zer i jedynek w której kolejność elementów jest zgodna z kolejnością w pliku konfiguracyjnym. W tym przypadku komunikacja odbywa się na porcie 9002.

Uwagi:

- Aplikacja powinna działać w systemach Windows i Linux
- Wskazane użycie jak najprostszych bibliotek np. TKinter
- Program powinien sygnalizować elementarne błędy:

  - element poza obszarem okna
  - błąd składni w pliku konfiguracyjnym
  - długość pakietu niezgodna z liczbą ledów