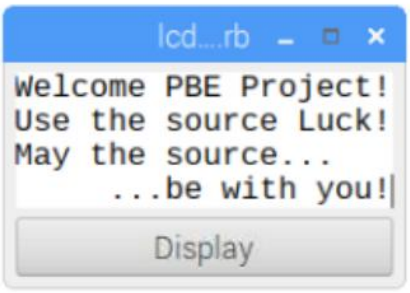
Bernat Ollé Ferrayuoli
3B

MEMORIA PUZZLE 2: LCD

OBJECTIUS

El objectiu principal d'aquest Puzzle 2 és programar una versió gràfica del Puzzle 1.



CONFIGURACIÓ DE VNC VIEWER

Per visualitzar la Raspberry Pi en un entorn gràfic és necessari utilitzar VNC Viewer. Aquest programa de software lliure (Virtual Network Computing) permet mostrar i controlar el contingut de la pantalla del sistema desitjar en un altre sistema, en el meu cas Raspian i Windows respectivament. Per tant, per instal·lar VNC Viewer cal accedir a la pàgina web oficial https://www.realvnc.com/es/connect/download/viewer/. Un cop el programa està instal·lat per accedir a la Raspberry hem d'introduir la seva ip, el mateix que fèiem al Puzzle 1 des de la terminal.

LLIBRERIES

Per poder realitzar la versió gràfica s'ha d'instal·lar una llibreria molt potent anomenada GTK 3, que s'utilitza com una aplicació web, amb CSS. Per instal·lar la llibreria s'ha d'introduir el següent comando:

```
- sudo apt-get install libgtk-3-dev
```

Per utilitzar CSS no es necessari instal·lar cap llibreria.

PROBLEMES

El únic problema que he tingut en aquest Puzzle 2 ha sigut que a la hora de connectar la raspberry al VNC Viewer utilitzava la notació "raspberrypi.local" en comptes de la direcció ip expressada amb números. Però em sortia un error, que ho vaig solucionar introduint directament la ip.

A més, no ho considero un problema per realitzar el puzzle 2 ja que no afectava directament al funcionament del projecte, quan visualitzava la raspberry a traves de VNC Viewer la resolució era molt baixa. Per solucionar el problema vaig entrar a la configuració de la raspberry i vaig canviar la resolució a la de la meva pantalla del portàtil.

CODI

```
import gi
gi.require_version("Gtk", "3.0")
from puzzle1 import LCD  #importem el puzzle 1
from gi.repository import Gtk, Pango, Gdk #importem llibreries de GTK3

class TextViewWindow(Gtk.Window):
    def __init__(self):
        Gtk.Window.__init__(self, title="Puzzle 2 Bernat")#creem la finesrta
        self.set_resizable(False)#fixem la finestra
        self.set_size_request(364,170)#posem la mida que volem
        self.box1 = Gtk.Box(orientation="vertical", spacing=4)#creem el box
        self.add(self.box1)

        self.textview = Gtk.TextView()# creem el TextView
        self.textbuffer = self.textview.get_buffer()#guardem el text introduit
        self.box1.pack_start(self.textview,True,True,0)#posem el TextView al box

        button = Gtk.Button(label="Display")#creem el botó
        button.connect("clicked", self.on_print_clicked)# asignem al boto la funció
correspondent
        button.set_name("button")#li poso un nom
        self.box1.pack_start(button,False,False,0)#l'afegeixo al box

        self.blue = b"""#creem els CSS per personalitzar la finestrea

        #button{
            background-color:#29c8f6;
            font-family:monospace;
            color: #FFFFFF;
            }

        textview{
            font:  30px monospace;
         }

        """
        self.css_provider = Gtk.CssProvider()
        self.css_provider.load_from_data(self.blue)
        self.context = Gtk.StyleContext()
        self.screen = Gdk.Screen.get_default()
        self.context.add_provider_for_screen(self.screen, self.css_provider ,
Gtk.STYLE_PROVIDER_PRIORITY_APPLICATION)#introduim el codi CSS a python

    def on_print_clicked(self, widget):
        l=LCD()#introduim el puzzle 1
        start = self.textbuffer.get_start_iter()
        end = self.textbuffer.get_end_iter()
        l.lcd_multiline(self.textbuffer.get_text(start,end,0))#recorro el text que
hi ha al text view i el mostrem al LCD
```

```
        self.textbuffer.set_text("")

win = TextViewWindow()
win.connect("destroy", Gtk.main_quit)
win.show_all()
Gtk.main()
```

Com es pot observar, primer inicialitzem les llibreries, seguidament creem la finestra, que la fixem a una mida predeterminada, i hi afegim un box a dins d'aquesta. Dins el box afegim el TextView, que és la principal funció d'aquest puzzle 2, també ens guardem el que s'escriu al TextView per després poder-ho mostrar al LCD. Finalment afegim un botó al box per poder decidir quan volem que el text introduït es mostri al LCD. Un cop creat tots els elements ens falta personalitzar cada un d'aquests components utilitzant CSS, el primer que modifiquem és la lletra que es mostra en el TextView que la posem monospace així no cada un del caràcter te la mateixa longitud, seguidament també canviem la font el botó a és del seu color de fons, que el posem blau. La última funció que apareix al codi anomenada "`on_print_clicked`" és la funció que es crida quan prenem el botó per mostrar el text al LCD, aquesta funció el que fa és cridar la funció de imprimir el string multi línia al LCD amb el text del buffer del Text View.