PUZZLE 1

- ## Objectiu:

L'objectiu d'aquest puzzle és ser capaç de fer una versió gràfica del puzzle 1 amb les biblioteques "PYGObject" on ens permet crear una finestra que quan llegeix la targeta ens mostrarà el seu "uid" en la finestra.

- ## Correcció codi del puzzle 1:

En l'anterior puzzle no vaig fer bé el codi llavors vaig fer correccions per ja aconseguir l'objectiu del primer puzzle. El primer error que vaig cometre va ser buscar llibreries per el meu perifèric, quan, en realitat, no és necessari aplicar cap llibreria en el meu codi. Com el lector RFID amb cable USB actua com un teclat, l'únic que haig de fer és llegir el que s'escriu en el terminal i després converteix-ho el que he llegit en hexadecimal.

RFIDreader.py > uid_hex

```
while True:
    # Llegeixo l'entrada del teclat
    uid = input("Ingrese tarjeta: ")
    uid_number = int(uid)  # Convierteix el string en numero
    uid_hex = hex(uid_number)[2:].upper()  # Convierteix el numero en hexadecimal i en mayusucules
```

- ## Llibreries:

Les llibreries que he utilitzat en aquest puzzle és:

- **"gi":** Aquesta llibreria conté totes les propietats objecte que necessito per poder crear finestres, etiquetes, botons, etc.

```
import gi

gi.require_version("Gtk", "3.0")
from gi.repository import GLib
from gi.repository import Gtk, Gdk
```

- **Codi:**

```python
class LabelWindow(Gtk.Window):
    def __init__(self):
        super().__init__(title="Lector de tarjeta")
        self.set_border_width(10)
        self.set_default_size(400, 200)

        # Crear el "box" on ficarem un "label" i un "button"
        vbox = Gtk.Box(orientation=Gtk.Orientation.VERTICAL, spacing=3)
        vbox.set_homogeneous(False)

        # Creem el "label"
        self.label = Gtk.Label(label="Please, login with your University class")
        self.label.override_background_color(Gtk.StateType.NORMAL, Gdk.RGBA(0, 0, 1, 1))
        vbox.pack_start(self.label, True, True, 0)

        # Creem l'entrada de text
        self.entry = Gtk.Entry()
        self.entry.set_text("")
        vbox.pack_start(self.entry, True, True, 0)

        # Creem un box horitzontal per ficar els botons
        hbox = Gtk.Box(spacing=2)
        hbox.set_homogeneous(False)

        # Creem el boto per enviar
        button_send = Gtk.Button(label="Send")
        button_send.connect("clicked", self.on_button_send_clicked)
        hbox.pack_start(button_send, True, True, 0)
        # Creem el boto per netejar el uid
        button_clear = Gtk.Button(label="Clear")
        button_clear.connect("clicked", self.on_button_clear_clicked)
        hbox.pack_start(button_clear, True, True, 0)

        #Afegim en la finestra el "box" creat
        vbox.pack_start(hbox, True, True, 0)
        self.add(vbox)


    #Aquesta funcio s'aplica quan el boto de netejar es prem
    def on_button_clear_clicked(self, widget):
        #Tornem visible un altre vegada l'entrada de text
        self.entry.set_text("")
        self.entry.set_visible(True)
        #Tornem el "label" en el seu estat inicial
        self.label.set_text("Please, login with your University class")
        self.label.override_background_color(Gtk.StateType.NORMAL, Gdk.RGBA(0, 0, 1, 1))

    def on_button_send_clicked(self, widget):
        uid = self.entry.get_text()
        uid_number = int(uid)  # Convierteix el string en numero
        uid_hex = hex(uid_number)[2:].upper()  # Convierteix el numero en hexadecimal i en mayusucules

        #Inicia la funcio que actualitza els objectes de la finestra
        self.update(uid_hex)
    def update(self, uid_hex):
        # Actualitzem el "label" amb el uid llegit
        self.label.set_text("UID: " + uid_hex)
        self.label.override_background_color(Gtk.StateType.NORMAL, Gdk.RGBA(1, 0, 0, 1))
        #amaga l'entrada de text
        self.entry.set_visible(False)



# Creem la finestra i ejecutem el "main" de Gtk
window = LabelWindow()
window.connect("destroy", Gtk.main_quit)
window.show_all()
Gtk.main()
```

En el codi el que he fet es crear quatre objectes a la finestra:

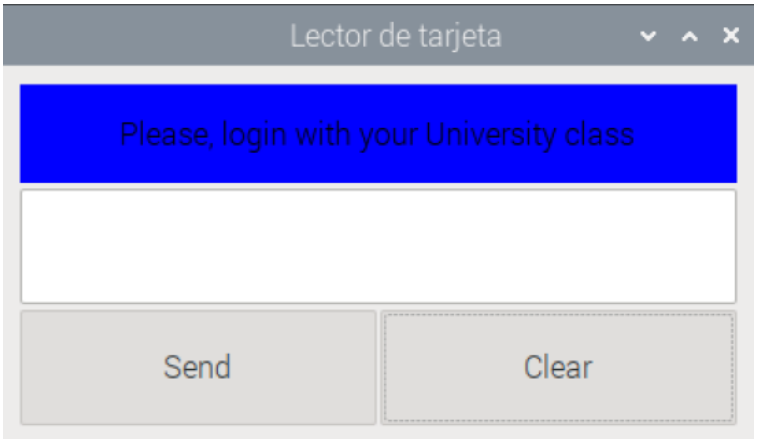
- **“Label”:** Es una etiqueta on s’actualitzarà per ficar el uid en hexadecimal de la targeta
- **“Entry”:** Es una entrada de text que serveix per escriure el uid de la targeta
- **“Button_clear”:** Es un boto que serveix per tornar en l’estat inicial de la finestra
- **“Button_send”:** Serveix per enviar el que s’ha llegit de la targeta en el enter.

Despres el següent que fa es esperar que s’escrigui en el “Entry” i despres quan esta escrit quan es prem el botó de “send” s’actualitza la finestra ficant el “uid” en hexadecimal en el label i canviant el fons del label en color vermell.

Quan es prem el botó clear es torna a l’estat inicial de la finestra.

Demostració del codi:

Abans de posar la targeta:



Després de posar la targeta:

- ## Problemes de codi:

  Al principi havia fet el codi utilitzant codi recurrent, el problema es que no podia aplicar el mateix codi que havia fet el puzzle 1 sense escriure desde terminal. És a dir, que no podia llegir la targeta sense tenir que recurrir a terminal.

  Solució:

  La solució ha sigut posar un entrada de text en la finestra i un botó que quan es prem s'envia l'"uid" en hexadecimal i s'actualitza la finestra. Amb aquesta solució no haig de recurrir a terminal ni tampoc en codi recurrent i segueix la mateixa idea del puzzle 1 respecte el perifèric.