Memòria PUZZLE 2

Al primer puzzle, es va desenvolupar una classe amb un mètode que permet llegir un lector de targetes - en el meu cas, el lector és el *Itead PN532 NFC*- i retorna un número que correspon a l'identificador d'usuari de la targeta o el clauer detectat. El segon puzzle consisteix a crear la versió gràfica d'aquest programa. Es pretén crear una finestra que mostri un text on es demana la identificació amb la targeta de la universitat. Un cop detectada la targeta - mitjançant el mètode desenvolupat anteriorment - apareix un nou text que mostra el seu identificador d'usuari. En tot moment, a la part inferior de la finestra ha d'aparèixer un botó amb la paraula "Clear" que permet retornar a l'estat de detecció d'una nova targeta.

- **Instal·lació llibreria Gtk3:**

Per tal de crear aquesta interfície gràfica, es fa ús de *'Gtk3'*, una biblioteca de components gràfics multiplataforma per a desenvolupar interfícies gràfiques d'usuari (GUI). Ja que em pertoca programar el codi en Ruby, utilitzem el mòdul *ruby-gnome2 Gtk3*.

Per tal d'instal·lar la biblioteca, cal escriure la següent comanda a la terminal:

```
gem install gtk3
```

Primerament, la gemma '*gtk3'* no s'instal·la correctament, i em retorna errors en obtenir certs fitxers. Per tant, executo la següent comanda a la terminal:

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

que fa referència al paquet necessari per a instal·lar la biblioteca.

Així i tot, en tornar a intentar instal·lar '*gtk3'*, em retorna un nou error, indicant que no reconeix *'rake'* i, per tant, em cal introduir la comanda: `gem install rake` per a instal·lar la gemma.

Finalment, ja puc executar a la terminal: `gem install gtk3` i m'indica que la biblioteca s'ha instal·lat correctament, me n'asseguro posant: `ruby list` i comprovant que apareix '*gtk3'* a la llista.

- **Desenvolupament del codi:**

Per tal d'aconseguir la interfície gràfica que es vol, a part d'utilitzar les funcions que proporciona la biblioteca Gtk3, faig servir CSS, un llenguatge que serveix per a personalitzar l'aspecte visual i el disseny, normalment, d'una pàgina web.

- Fitxer CSS:

En primer lloc creo un fitxer tipus CSS (l'anomeno 'fitxer.css') on defineixo el color de fons i altres característiques estètiques de la finestra i els elements que té.

De l'element *'my_label'*, especifico el color de fons (blau) la mida i color del text i faig que el seu requadre tingui les puntes arrodonides. Creo una variant d'aquest element ('red') on només canvia el color de fons (vermell) que caldrà més endavant.

De l'element *'my_button'*, especifico diverses característiques; entre elles el color de fons (gris). Determino que quan el cursor es troba sobre el botó, el color de fons passa a ser un gris més fosc.

Finalment, de *'my_window'* només defineixo el color de fons (blanc).

```css
/* fitxer.css */

#my_label {
  background-color: #84DBFA; /* Fons blau */
  font-size: 16px;
  color: black;
  border-radius: 10px;
}

#my_label.red {
  background-color: #ED77B6; /* Fons vermell */
}

#my_button {
  background-color: #BDBDBD; /* Botó gris */
  color: black;
  border: black;
  padding: 10px 10px;
  text-decoration: none;
  font-size: 16px;
  margin: 2px 2px;
  border-radius: 10px;
}

#my_button:hover {
   background-color: #969696; /* Gris més fosc quan passes per sobre amb el cursor */
}

#my_window {
   background-color: #FFFFFF; /* Fons blanc */
}
```

- Codi:

Un cop descrit el fitxer CSS, desenvolupo el codi per obtenir el programa desitjat. Aquest consisteix en la creació d'una finestra (**window**), on tenim un text a la part superior (**label**) i un botó de 'Clear' a la part inferior (**button**). Inicialment, el text que es mostra és *"Please, login with your university card"*, mentre el programa es manté en espera de detectar una targeta (funció **read_uid** de la classe **Rfid**). Un cop detectada, mostra el seu identificador d'usuari per pantalla i canvia el color de fons. Si es prem el botó de *'Clear'* es retorna a la situació inicial en què demana que t'identifiquis amb la teva targeta de la universitat.

```ruby

require 'gtk3'
require_relative 'puzzle1'

def llegir_uid(label, rfid)
  label.style_context.remove_class("red")
  label.text = "Please, login with your university card"

  Thread.new do
    uid = rfid.read_uid

    GLib::Idle.add do
      label.style_context.add_class("red")
      label.text = "Uid: " + uid
      false
    end
  end
end
```

En primer lloc, s'importa la llibreria *'gtk3'* i el fitxer *'puzzle1'* on està definida la classe **Rfid** i el seu mètode per a detectar targetes.

A continuació, definim la funció **llegir_uid(label, rfid)**, aquesta funció mostra el text *"Please, login with your university card"* i restableix el seu color de fons a blau. Després crea un nou thread on cridar a la funció **read_uid**, ja que aquesta és bloquejant, i sinó la interfície gràfica es quedaria congelada mentre s'espera la lectura de targeta. Un cop obtingut l'**uid**, s'utilitza **GLib::Idle.add** per actualitzar la interfície des del fil principal: es canvia el text anterior de **label** per l'**uid** obtingut i es canvia el color de fons a vermell.

```ruby

window = Gtk::Window.new("Puzzle 2")
window.set_size_request(500, 1500)
window.signal_connect("destroy") { Gtk.main_quit }
window.set_name("my_window")

rfid1 = Rfid.new()

vbox = Gtk::Box.new(:vertical, 2)

label = Gtk::Label.new("Please, login with your university card")
label.set_name("my_label")
vbox.pack_start(label, :expand => true, :fill => true, :padding => 5)

button = Gtk::Button.new(label: "Clear")
button.set_name("my_button")
vbox.pack_start(button, :expand => true, :fill => true, :padding => 5)
```

El programa principal crea una finestra; determina la seva mida mínima; i li aplica l'estilitzat de l'element *'my_window'* definit al fitxer CSS. Connecta el senyal de destrucció de la finestra a Gtk.main_quit de manera que s'acaba el programa si es tanca la finestra. També crea un objecte de la classe **Rfid**, per després utilitzar el seu mètode de lectura.

Tot seguit, crea un contenidor vertical, on afegeix **label** (text mostrat a la finestra) i **button** (botó de *'Clear'*). Estilitza ambdues variables amb els elements corresponents definits al fitxer CSS.

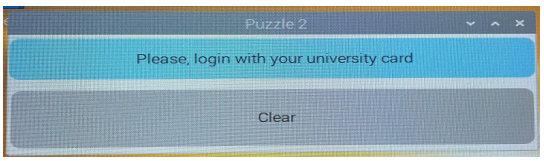
```
36
37 css_provider = Gtk::CssProvider.new
38 css_provider.load_from_path("fitxer.css")
39 Gtk::StyleContext.add_provider_for_screen(Gdk::Screen.default, css_provider, Gtk::StyleProvider::PRIORITY_USER)
40
41 window.add(vbox)
42 window.show_all
43
44 Gtk.main_iteration
45
46 llegir_uid(label, rfid1)
47
48 button.signal_connect("clicked") do
49   llegir_uid(label, rfid1)
50 end
51
52 Gtk.main
```

Aleshores, crea un proveïdor CSS i es carrega el fitxer d'estilitzat prèviament desenvolupat. Afegeix el contenidor vertical a la finestra i ho mostra tot; es crida a **Gtk.main_iteration** per a poder processar esdeveniments de la GUI abans d'iniciar la lectura de l'uid.

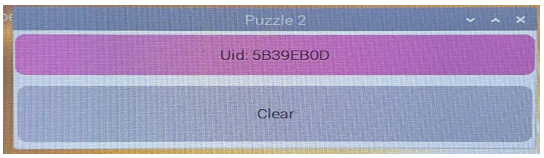
Es fa una primera crida a la funció de lectura de l'uid, i seguidament, es tornarà a cridar a la funció cada cop que es premi el botó *'Clear'*. Finalment, s'engega el bucle principal de GTK, que manté l'aplicació en execució i espera que succeeixin esdeveniments.

- **Execució del programa:**

En executar el programa mitjançant la comanda `ruby puzzle2.rb` a la terminal, apareix la finestra.



En apropar una targeta al lector, mostra el seu UID.



Si se sitúa el cursor sobre el botó, aquest canvia de color, i si es prem, es retorna a la situació inicial: