

Memòria puzzle 2

Pol Rodríguez Belló

PBE

1. Instal·lació de Ruby-GNOME/GTK3

Aquesta biblioteca permet crear una interfàç gràfica en Ruby. Permet crear finestres, botons (que poden detectar el click), entrades de text, quadres de diàleg...

Per fer-ho he posat les següents ordres al terminal de la RaspberryPi:

Unset

```
sudo apt update  
sudo apt install -y libgtk-3-dev gir1.2-gtk-3.0  
gem install gtk3
```

2. Codi

Primer de tot vaig modificar el codi del primer puzzle per a poder utilitzar-lo en aquest:

```
Unset
require 'mfrc522'

module LectorID
  def llegir_uid
    lector = MFRC522.new

    begin
      lector.picc_request(MFRC522::PICC_REQA)
      uid = lector.picc_select

      if uid.is_a?(Array) && !uid.empty?
        uid_1 = uid[0]
        uid_hex = uid_1.map { |byte| "%02X" % byte}.join
        return uid_hex
      else
        return nil
      end
    rescue => e
      puts "Error: #{e.message}"
      return nil
    end
  end
end
```

Com es pot observar, he definit el codi com un mòdul que té el mètode `llegir_uid`. Per altre banda també he modificat el codi per a que no imprimeixi cap text en el terminal.

A continuació, hi ha el codi del segon puzzle:

```
Unset
require_relative 'primer_puzzle'
require 'gtk3'
require 'thread'

class RFID
  def inicialitzacio
    @window = Gtk::Window.new("rfid_gtk.py")
```

```

@window.set_size_request(300, 150)
@window.signal_connect("destroy"){Gtk.main_quit}

@vbox = Gtk::Box.new(:vertical, 10)

@label = Gtk::Label.new("Si us plau apropa el teu carnet UPC")
@label.override_background_color(:normal, Gdk::RGBA.new(0, 0, 1, 1))
@label.override_color(:normal, Gdk::RGBA.new(1, 1, 1, 1))

@button = Gtk::Button.new(label: "Neteja")
@button.signal_connect("clicked") {neteja_etiqueta}

@vbox.pack_start(@label, expand: true, fill: true, padding: 5)
@vbox.pack_start(@button, expand: false, fill: false, padding: 5)

@window.add(@vbox)
@window.show_all

inici_rfid_fil
end

def neteja_etiqueta
  @label.set_text("Si us plau apropa el teu carnet UPC")
  @label.override_background_color(:normal, Gdk::RGBA.new(0, 0, 1, 1))
end

def inici_rfid_fil
  Thread.new do
    loop do
      begin
        uid = LectorID.llegir_uid
        if uid
          actualitza_etiqueta("uid: #{uid}", :red)
        else
          actualitza_etiqueta("No s'ha detectat la
targeta", :blue)
        end
      rescue => e
        actualitza_etiqueta("Error: #{e.message}", :red)
      end
      sleep 2
    end
  end
end

def actualitza_etiqueta(text, color)
  GLib::Idle.add do
    @label.set_text(text)
    bg_color = color == :red ? Gdk::RGBA.new(1, 0, 0, 1) :
Gdk::RGBA.new(0, 0, 1, 1)
    @label.override_background_color(:normal, bg_color)
    true
  end
end

```

```
end

Gtk.init
RFID.new
Gtk.main
```

Vaig provar aquest codi i no funciona, ja que quan s'executa el terminal es queda en espera fins que apareix un codi d'error perquè s'ha arribat al límit de temps d'espera.

Per altre banda, el que hauria de fer aquest codi es podria separar en x parts:

1. Creació de la classe RFID
2. Creació de la finestra principal, amb el text de "rfid_gtk.py" com a nom de la finestra i amb una mida determinada.
3. Personalització de la finestra, amb l'etiqueta "Si us plau apropa el teu carnet UPC" i posant el fons de color blau. A més, la inicialització d'un botó que permet netejar la finestra amb un funció definit més avall.
4. Afegir tots els elements al contenidor i mostrar tot per pantalla.
5. Funció neteja_etiqueta que torna a mostrar el text i el color inicial.
6. Fil que llegeix la targeta amb un bucle cada 2 segons.
7. Funció actualitza_etiqueta permet canviar el text i el color de la finestra.
8. Inicialització del programa.