Article **Journal of Scientific and Technical Applications**
June 2022, Vol.8 No.21 10-15

Application to monitoring a USB control with Ruby in Windows and Linux Ubuntu

Aplicación para monitoreo de un control USB con Ruby en Windows y Linux Ubuntu

ESPARZA-CASTILLO, Ramón Ángel†*, LÓPEZ-ROMO, José Alonso, MEZA-IBARRA, Iván Dostoyewski and ABRIL-GARCÍA, José Humberto

Universidad Tecnológica de Hermosillo, Information Technologies Engineering, Mexico.

ID 1st author: *Ramon Angel, Esparza-Castillo* / **ORC ID:** 0000-0001-6929-8798, **Researcher ID Thomson:** 3453138, **arXiv ID:** Ramon.Esparza, **CVU CONACYT ID:** 1070055

ID 1st Coauthor: *José Alonso, Lopez-Romo* / **ORC ID:** 0000-0001-7428-1480, **Researcher ID Thomson:** R-5616-2018, **arXiv**: alonsolopez2, **CVU CONACYT ID:** 944227

ID 2nd Coauthor: *Iván Dostoyewski, Meza-Ibarra* / **ORC ID:** 0000-0001-6139-032X, **Researcher ID Thomson:** F-3550-2018, **arXiv**: imeza, **CVU CONACYT ID:** 769494

ID 3rd Coauthor: *José Humberto, Abril-García* / **ORC ID:** 0000-0003-3494-6817, **Researcher ID Thomson:** F-4252-2018, **arXiv ID**: jhabril, **CVU CONACYT ID:** 204935

DOI: 10.35429/JSTA.2022.21.8.10.15 Received January 10, 2022; Acceptance February 28, 2022

Abstract

This work shows the development of an application in Ruby to detect the actions carried out on a USBNes control, with the purpose of achieving an application that can be used for teaching video game programming, connectivity and communication between devices, in the Ruby programming language. In the first stage, an analysis was made of the libraries available in Ruby such as Shoes and Gosu to create GUIs and video games and the detection of an USB device. In the second stage two prototypes were identified that were able to detect the actions carried out on the keyboard. In the last stage the final version was developed which detects the buttons pressed on the remote, the necessary tests were made, and the code was published in a repository in GITLAB for reference and future use. The application was developed and tested for Windows and Linux Ubuntu to prove Ruby's portability.

Ruby, Shoes, Gosu, USBNes

Resumen

El presente trabajo muestra el desarrollo de una aplicación en Ruby para detectar las acciones realizadas sobre un control USBNes, con el objetivo de lograr una aplicación que puede ser usada para la enseñanza de programación de videojuegos, y la conectividad y la comunicación entre dispositivos, en Ruby. En la primera etapa se hizo un análisis de las librerías disponibles en Ruby como Shoes y Gosu para crear GUI y videojuegos y la detección del dispositivo USB, en la segunda etapa se desarrollaron dos prototipos que detectan las acciones realizadas en el teclado, en la tercera etapa se desarrolló la versión final la cual detecta los botones presionados en el mando. Se hicieron las pruebas necesarias y se publicó el código en un repositorio en GITLAB para referencia y uso futuro. La aplicación fue desarrollada para Windows y Linux Ubuntu para probar la portabilidad de Ruby.

Ruby, Shoes, Gosu, USBNes

Citation: ESPARZA-CASTILLO, Ramón Ángel, LÓPEZ-ROMO, José Alonso, MEZA-IBARRA, Iván Dostoyewski and ABRIL-GARCÍA, José Humberto. Application to monitoring a USB control with Ruby in Windows and Linux Ubuntu. Journal of Scientific and Technical Applications. 2022. 8-21:10-15.

* Correspondence to Author (Email: tic17311110@uthermosillo.edu.mx)
† Researcher contributing as first author.

Introduction

The commerce and development of video games is one of the most lucrative industries, above cinema and television, for this reason it is of great importance to know the basics of video game programming for at least one development technology, so we consider it relevant to propose the development of an application that displays a GUI where a USBNes control is visualised and that displays on screen the buttons and pad that are pressed on the control. For the development of this project we use Visual Studio Code and the Shoes and Gosu libraries for Ruby. The result is several prototypes that can be used as a basis for further development in various areas of engineering such as GUI application development, video games or IoT. Figure 1 shows the general diagram of the project.

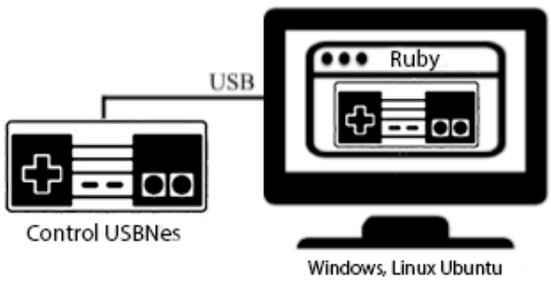

Figure 1 Project overview diagram

Tools

Ruby (Cooper, 2007) (Mahadevan, 2002) is an interpreted, object-oriented programming language created by Yukihiro Matsumoto, who started working on Ruby in 1993 and introduced it in 1995. It combines syntax inspired by Python and Perl with object-oriented programming features. Ruby is distributed under a free software licence.

Shoes (Gandy, 2020) (Coupe, 2022) is a framework of tools based on the Ruby programming language. Shoes runs on Microsoft Windows, Mac OS X and Linux (GTK+), using the underlying Cairo and Pango technologies.

Gosu (Sobkowicz, 2015) (Julian Raschke, 2020) is a 2D game development library for Ruby and C++. It is available for macOS, Windows, Linux (including Raspbian) and iOS. Gosu is lightweight and has few dependencies (mainly SDL 2). It provides a window and a main method, 2D graphics and text, OpenGL or OpenGL ES technology, sounds, music, keyboard, mouse and gamepad input.

Hardward and Software Specifications

Hardward used in the project

- Intel(R) Core (TM) i34025U CPU @ 1.90 GHz, 6 GB RAM, 64 bits.
- Pentium (R) Dual-Core CPU E5700 @ 3.00 GHz, 4 GB RAM, 32 bits.
- Game Controller ELE-GATE® GM.09

Software used in the project Windows 10

- Ruby 2.7.1p83 (2020-03-31 revision a0c7c23c9c) [x64-mingw32]
- Shoes federales build 3.2.25 r2170
- Shoes walkabout build 3.3.7 r3301
- Gosu (0.15.2 x64-mingw32)
- Visual Studio Code version 1.45.1

Linux Ubuntu 20.04 LTS

- Ruby 2.7.0p0 (2019-12-25 revision 647ee6f091) [x86_64-linux-gnu]
- Shoes federales build 3.2.25 r2170
- Shoes walkabout build 3.3.7 r3301
- Gosu (0.15.2)
- Visual Studio Code version 1.46.1

Development

In order to evaluate the development of graphical interfaces with Ruby and communication with USB ports, several prototypes were developed.

ISSN 2444-4928

Figure 2 shows a GUI developed in Shoes federal for Windows 10 and Figure 3 shows a GUI developed in Shoes walkabout. These prototypes allow to identify the graphical recognition in the GUI through the keyboard, as it simulates the buttons on the control shown in the application.

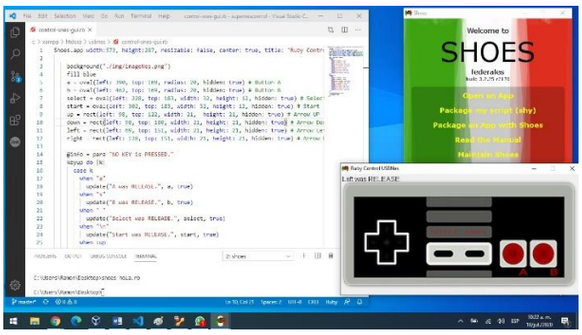

Figure 2 Federal Windows Shoes

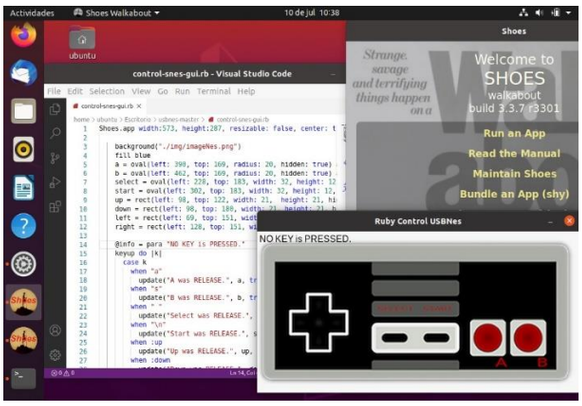

Figure 3 Linux shoes walkabout

Figure 4 shows a Ruby GUI running on Windows with the Gosu library.

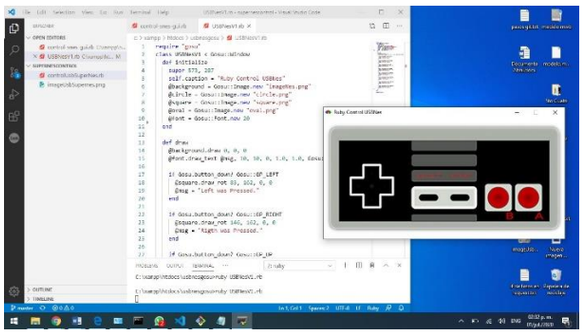

Figure 4 Windows Gosu

Figure 5 shows a Ruby GUI running on Linux Ubuntu with the Gosu library.

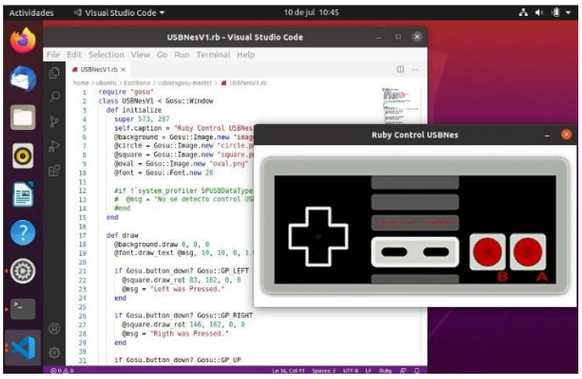

Figure 5 Linux Ubuntu Gosu

After developing and testing the prototypes, it was decided to choose Gosu for the final development of the project, because it integrates directly both the GUI development and the access to the USB ports. In the final stage of the project we integrated the control for the recognition of the buttons and the pad showing an image when they are pressed.

Results and Conclusions

Figure 6 shows the final result of our project, where the USBNes controller (Game Controller ELE-GATE GM.09) is connected and operating with the visual interface.

With the development of this work we managed to generate an application of recognition of a USBNes control compatible with Windows and Linux Ubuntu operating systems, using the Ruby programming language which serves as a guide to dabble in the development and creation of video games in different operating systems, where students can get many of the benefits of this development language, among which we can mention the ease of distribution, integration of tools and libraries to design, program and test our video games.

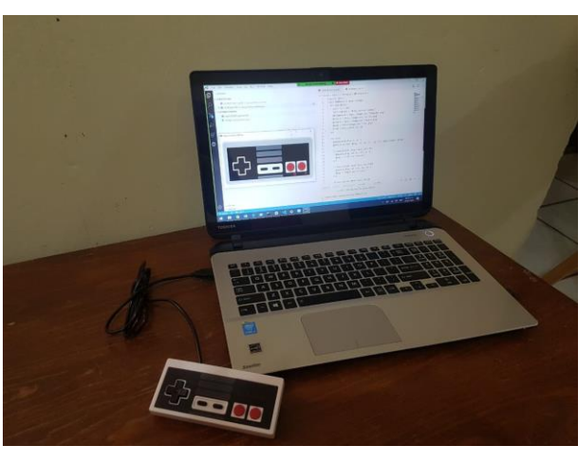

Figure 6

ISSN 2444-4928

Annexes

Application code for Windows and Linux Ubuntu with Shoes. (This code is not registered on any authoring site).

```
Shoes.app width:573, height:287, resizable: false, center: true, title: "Ruby Control USBNes" do

  background("./img/imageNes.png")

  fill blue

  a = oval(left: 390, top: 169, radius: 20, hidden: true) # Button A

  b = oval(left: 462, top: 169, radius: 20, hidden: true) # Button B

  select = oval(left: 228, top: 183, width: 32, height: 12, hidden: true) # Select

  start = oval(left: 302, top: 183, width: 32, height: 12, hidden: true) # Start

  up = rect(left: 98, top: 122, width: 21,  height: 21, hidden: true) # Arrow UP

  down = rect(left: 98, top: 180, width: 21, height: 21, hidden: true) # Arrow Down

  left = rect(left: 69, top: 151, width: 21, height: 21, hidden: true) # Arrow Left

  right = rect(left: 128, top: 151, width: 21, height: 21, hidden: true) # Arrow Right

  @info = para "NO KEY is PRESSED."

  keyup do |k|

    case k

      when "a"

        update("A was RELEASE.", a, true)

      when "s"

        update("B was RELEASE.", b, true)

      when " "

        update("Select was RELEASE.", select, true)

      when "\n"

        update("Start was RELEASE.", start, true)

      when :up

        update("Up was RELEASE.", up, true)

      when :down

        update("Down was RELEASE.", down, true)

      when :left

        update("Left was RELEASE.", left, true)

      when :right

        update("Right was RELEASE.", right, true)

    end

  end

  keypress do |k|

    case k

      when "a"

        update("A was PRESSED.", a, false)

      when "s"

        update("B was PRESSED.", b, false)

      when " "

        update("Select was PRESSED.", select, false)

      when "\n"

        update("Start was PRESSED.", start, false)

      when :up

        update("Up was PRESSED.", up, false)

      when :down

        update("Down was PRESSED.", down, false)
```

```
      when :left
        update("Left was PRESSED.", left, false)
      when :right
        update("Right was PRESSED.", right, fal
se)
    end
  end
  def update(message, button, state)
    @info.replace message
    button.hidden = state
  end
end
```

Application Code for Windows and Linux Ubuntu with Gosu.

```
require "gosu"
class USBNesV1 < Gosu::Window
  def initialize
    super 573, 287
    self.caption = "Ruby Control USBNes"
    @background = Gosu::Image.new "imageNe
s.png"
    @circle = Gosu::Image.new "circle.png"
    @square = Gosu::Image.new "square.png"
    @oval = Gosu::Image.new "oval.png"
    @font = Gosu::Font.new 20
  end
  def draw
    @background.draw 0, 0, 0
    @font.draw_text @msg, 10, 10, 0, 1.0, 1.0,
Gosu::Color::BLACK
    if Gosu.button_down? Gosu::GP_LEFT
      @square.draw_rot 83, 162, 0, 0
      @msg = "Left was Pressed."
    end
    if Gosu.button_down? Gosu::GP_RIGHT
      @square.draw_rot 146, 162, 0, 0
      @msg = "Rigth was Pressed."
    end
    if Gosu.button_down? Gosu::GP_UP
      @square.draw_rot 114.5, 131, 0, 0
      @msg = "Up was Pressed."
    end
    if Gosu.button_down? Gosu::GP_DOWN
      @square.draw_rot 114.5, 193, 0, 0
      @msg = "Down was Pressed."
    end
    if Gosu.button_down? Gosu::GP_BUTTON_
3
      @circle.draw_rot 415, 190, 0, 0
      @msg = "B was Pressed."
    end
    if Gosu.button_down? Gosu::GP_BUTTON_
1
      @circle.draw_rot 487, 190, 0, 0
      @msg = "A was Pressed."
    end
    if Gosu.button_down? Gosu::GP_BUTTON_
4
      @oval.draw_rot 249, 189, 0, 0
      @msg = "Select was Pressed."
    end
```

ISSN 2444-4928

```
  if Gosu.button_down? Gosu::GP_BUTTON_
6

    @oval.draw_rot 324, 189, 0, 0

    @msg = "Start was Pressed."

  end

 end

end
```

USBNesV1.new.show

References

Cooper, P. (2007). *Beginning Ruby: From Novice to Professional.* New York: Apress.

Coupe, C. (2022, 02 09). *Walkabout Shoes | Keeping Red Shoes alive since January 2014*. Retrieved from https://github.com/shoes/shoes3

Gandy, D. (2020, 07 28). *Shoes! The easiest little GUI toolkit, for Ruby.* Retrieved from http://shoesrb.com/

Julian Raschke, J. L. (2020, 07 15). *Hello • Gosu*. Retrieved from https://www.libgosu.org

Mahadevan, S. (2002). *Making Use of RUBY.* Indianapolis, Indiana: Wiley Publishing, Inc.

Sobkowicz, M. (2015). *Learn Game Programming with Ruby Bring Your Ideas to Life with Gosu.* United States of America: The Pragmatic Bookshelf.

ISSN 2444-4928