SWINBURNE UNIVERSITY OF TECHNOLOGY

COS6006 Postgraduate
Introduction to Programming

Week 4
GUI – Supplement

Dr. Quang-Hung Luu – hluu@swin.edu.au
Dr. Matthew Mitchel – mmitchell@swin.edu.au
Swinburne University of Technology

Instruction

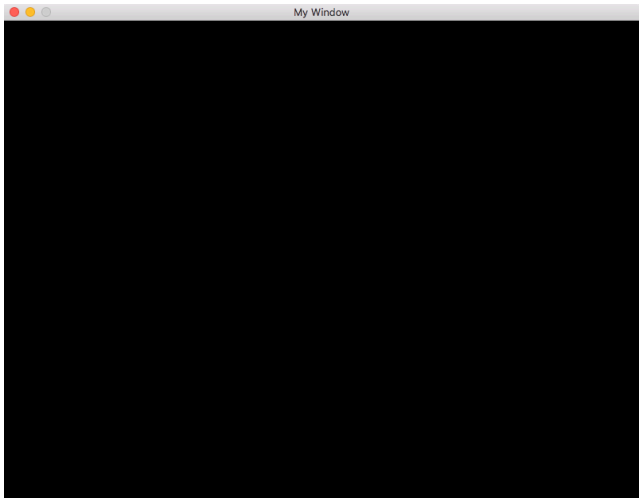
- To create a gosu-based application?
  - Gosu library consists of various functions/procedures to build Ruby applications with graphic user interface (GUI).
  - Used the blank Ruby template to start, as provided in next slide. (It was shown during Week 4 lecture as well).
- To draw a shape with gosu?
  - To draw a shape, we need to call its built-in function given in Gosu, and pass it the correct parameters and in correct order.
  - Follow the instructions in the slides for examples of drawing a quadratic shape, a triangle, a rectangular.

Demo > Empty gosu application

Your code stored in **mywindow.rb**

```ruby
# load gosu library
require 'gosu'

# define the "prototype" of your application
class MyWindow < Gosu::Window

    # initial values when application load
    def initialize()
        super 800,600,false
        self.caption = "My Window"
    end

    # put main logic of program inside update()
    def update()
    end

    # callback to handle inputs from user, such as mouse and keyboard
    def button_down(id)
    end

    # draw the interface inside draw
    def draw()
    end
end

# start your application by constructing an instance of prototype
window = MyWindow.new()
window.show()
```

What you see after running it in Terminal



This is the default Gosu template, you just copy it and run without changing anything.

Download source codes in this slide from:

Canvas > COS60006 > Modules > Lecture Materials > Week 4 > Supplement: supcodes.zip

Demo > Empty gosu application

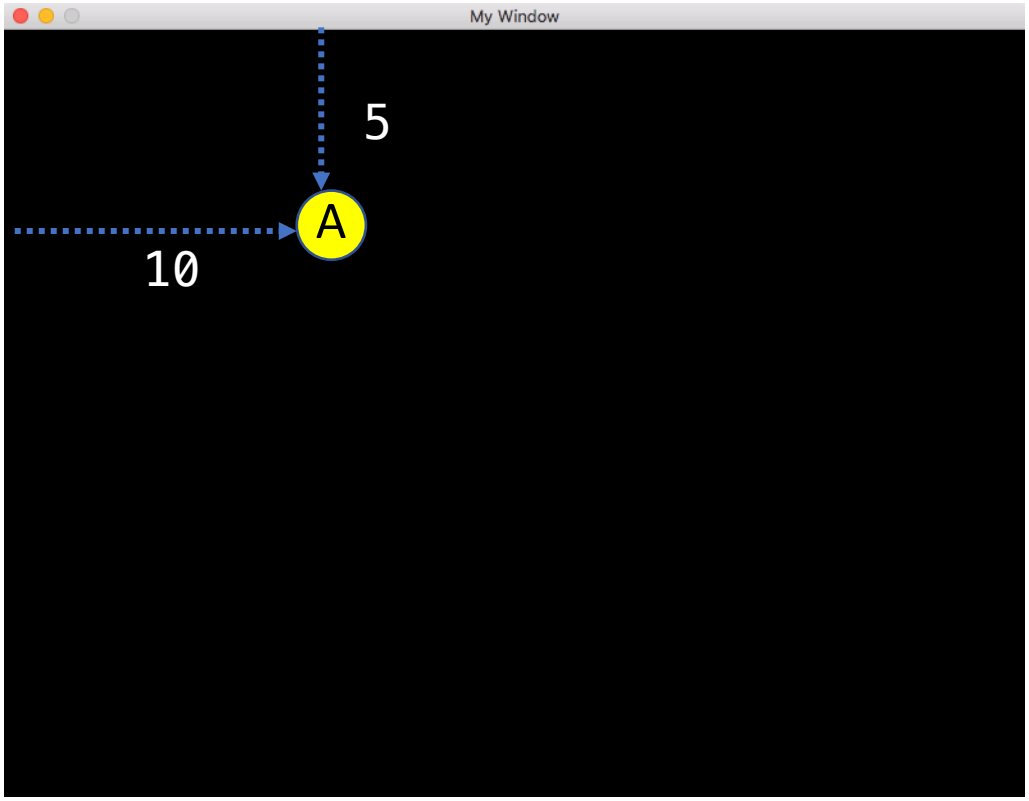
Your code stored in **mywindow.rb**

```
1  # load gosu library
2  require 'gosu'
3
4  # define the "prototype" of your application
5  class MyWindow < Gosu::Window
6
7      # initial values when application load
8      def initialize()
9          super 800,600,false
10         self.caption = "My Window"
11     end
12
13     # put main logic of program inside update()
14     def update()
15     end
16
17     # callback to handle inputs from user, such as mouse and keyboard
18     def button_down(id)
19     end
20
21     # draw the interface inside draw
22     def draw()
23     end
24 end
25
26 # start your application by constructing an instance of prototype
27 window = MyWindow.new()
28 window.show()
```

What you see after running it in Terminal

Demo > Coordinate and color of a point

The point A is often defined by three components: **X coordinate, Y coordinate, and Color.**



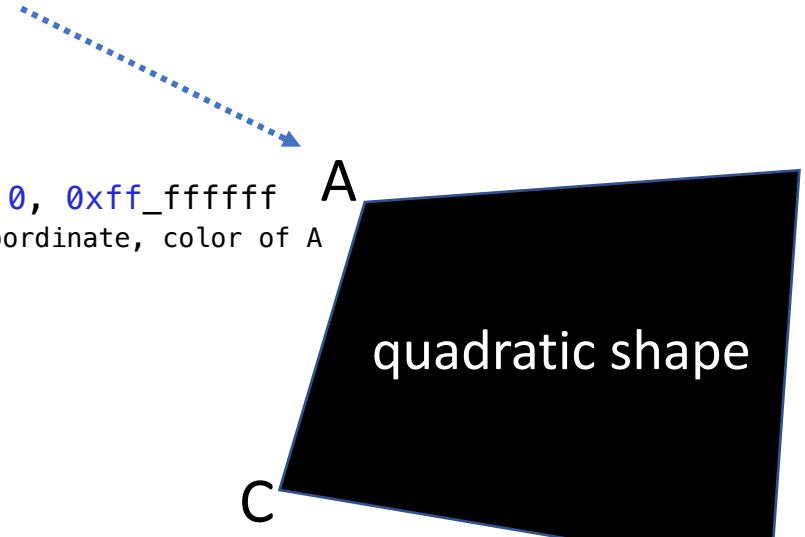
The origin of coordinate O is at the top left corner of window. X coordinate is horizontal, Y coordinate is vertical.

Demo > draw_quad

This function is to draw a quadratic shape, that is defined by 4 points.

```
draw_quad(0, 0, 0xff_ffffff, 640, 0, 0xff_ffffff, 0, 400, 0xff_ffffff, 640, 400, 0xff_ffffff, 1)
```

A B C D z coordinate



0, 0, 0xff_ffffff
x coordinate, y coordinate, color of A

640, 0, 0xff_fffff
x coordinate, y coordinate, color of B

0, 400, 0xff_ffffff
x coordinate, y coordinate, color of C

640, 400, 0xff_ffffff
x coordinate, y coordinate, color of D

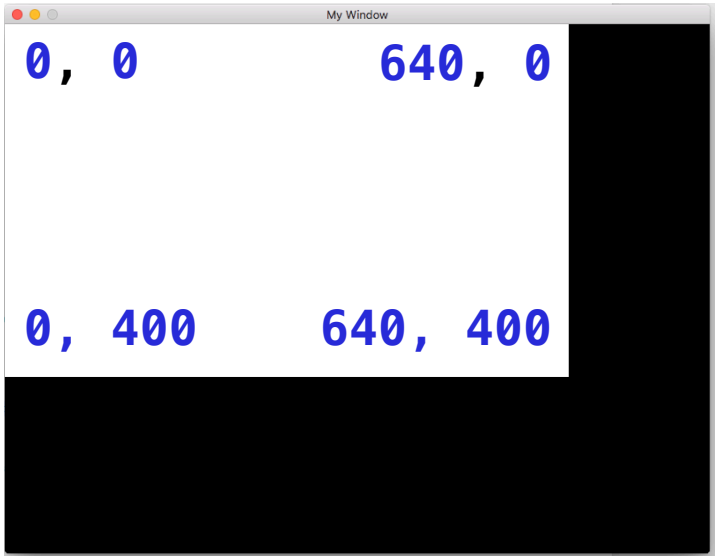
Demo > draw_quad

```
draw_quad(0, 0, 0xff_ffffff, 640, 0, 0xff_ffffff, 0, 400, 0xff_ffffff, 640, 400, 0xff_ffffff, 1)
```

Your code stored in **mywindow.rb**

```
1  # load gosu library
2  require 'gosu'
3
4  # define the "prototype" of your application
5  class MyWindow < Gosu::Window
6
7      # initial values when application load
8      def initialize()
9          super 800,600,false
10         self.caption = "My Window"
11     end
12
13     # put main logic of program inside update()
14     def update()
15     end
16
17     # callback to handle inputs from user, such as mouse and keyboard
18     def button_down(id)
19     end
20
21     # draw the interface inside draw
22     def draw()
23         draw_quad(0, 0, 0xff_ffffff, 640, 0, 0xff_ffffff, 0, 400, 0xff_ffffff, 640, 400, 0xff_ffffff, 1)
24     end
25 end
26
27 # start your application by constructing an instance of prototype
28 window = MyWindow.new()
29 window.show()
```

What you see after running it in Terminal



put your code for drawing here in the template

Demo > draw_quad

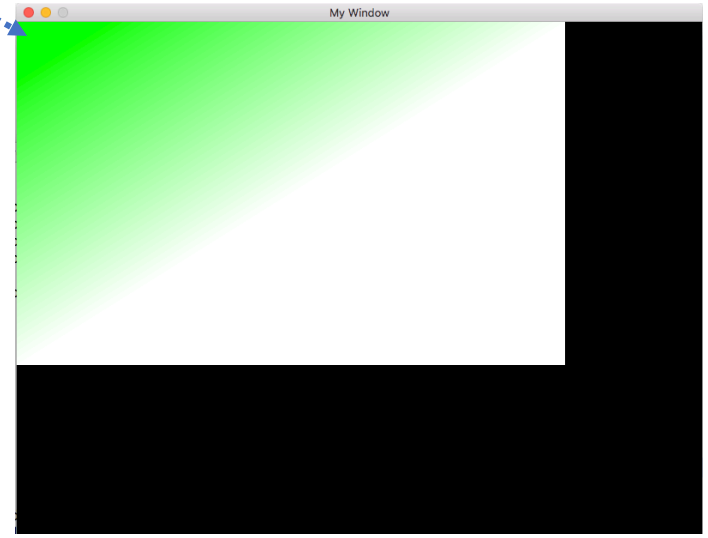
```
draw_quad(0, 0, Gosu::Color::GREEN, 640, 0, 0xff_ffffff, 0, 400, 0xff_ffffff, 640, 400, 0xff_ffffff, 1)
```

You can change the color at a corner of the shape

Your code stored in **mywindow.rb**

```
1  # load gosu library
2  require 'gosu'
3
4  # define the "prototype" of your application
5  class MyWindow < Gosu::Window
6
7      # initial values when application load
8      def initialize()
9          super 800,600,false
10         self.caption = "My Window"
11     end
12
13     # put main logic of program inside update()
14     def update()
15     end
16
17     # callback to handle inputs from user, such as mouse and keyboard
18     def button_down(id)
19     end
20
21     # draw the interface inside draw
22     def draw()
23         draw_quad(0, 0, Gosu::Color::GREEN, 640, 0, 0xff_ffffff, 0, 400, 0xff_ffffff, 640, 400, 0xff_ffffff, 1)
24     end
25 end
26
27 # start your application by constructing an instance of prototype
28 window = MyWindow.new()
29 window.show()
```

What you see after running it in Terminal



Demo > draw_triangle

This function is to draw a triangle,
that is defined by 3 points in the coordinates.

```
draw_triangle(50, 50, Gosu::Color::GREEN, 100, 50, Gosu::Color::GREEN, 50, 100, Gosu::Color::RED, 1,mode=:default)
```

A

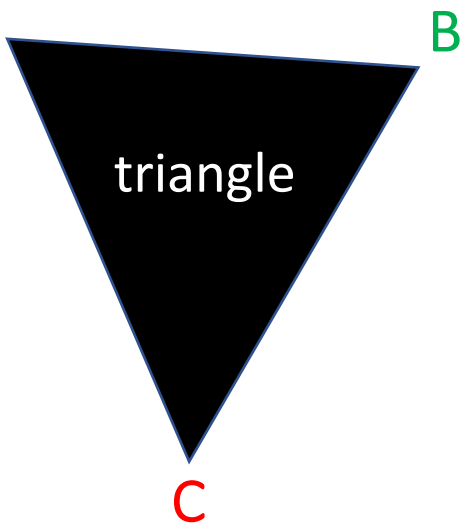
B

C

z location

use default setting of triangle

50, 50, Gosu::Color::GREEN
x coordinate, y coordinate, color of A



Gosu::Color::GREEN
x coordinate, y coordinate, color of B

50, 100, Gosu::Color::RED
x coordinate, y coordinate, color of C

Demo > draw_quad

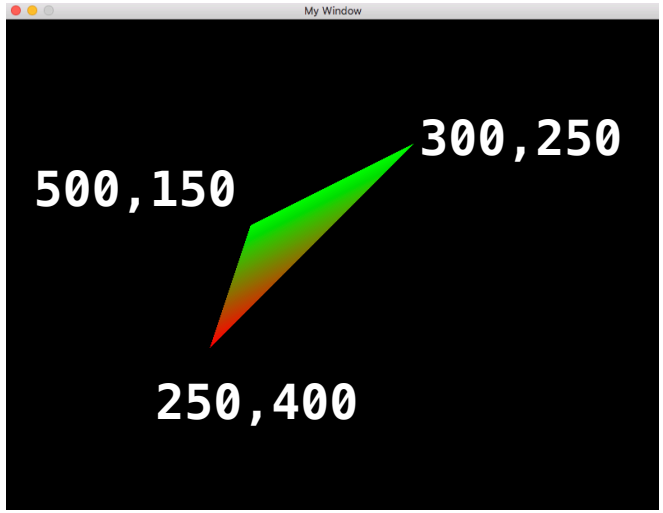
Draw a quadratic shape, i.e. a shape having 4 points

```
draw_triangle(500, 150, Gosu::Color::GREEN, 300, 250, Gosu::Color::GREEN, 250, 400, Gosu::Color::RED, 1, mode=:default)
```

Your code stored in **mywindow.rb**

```
1  # load gosu library
2  require 'gosu'
3
4  # define the "prototype" of your application
5  class MyWindow < Gosu::Window
6
7      # initial values when application load
8      def initialize()
9          super 800,600,false
10         self.caption = "My Window"
11     end
12
13     # put main logic of program inside update()
14     def update()
15     end
16
17     # callback to handle inputs from user, such as mouse and keyboard
18     def button_down(id)
19     end
20
21     # draw the interface inside draw
22     def draw()
23         draw_triangle(500, 150, Gosu::Color::GREEN, 300, 250, Gosu::Color::GREEN, 250, 400, Gosu::Color::RED, 1, mode=:default)
24     end
25 end
26
27 # start your application by constructing an instance of prototype
28 window = MyWindow.new()
29 window.show()
```

What you see after running it in Terminal



put your code for drawing here in the template

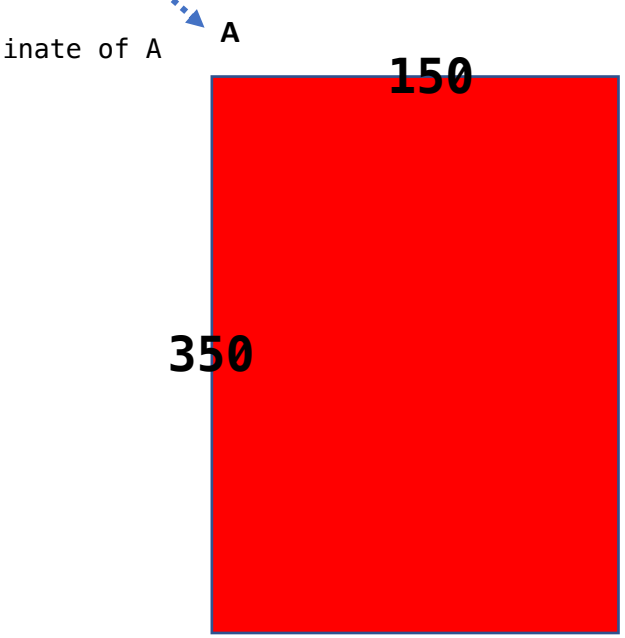
Demo > draw_rect

This function is to draw a rectangular,
that is defined by only 1 points and its sizes.

```
Gosu.draw_rect(300, 200, 150, 350, Gosu::Color::RED, 2, mode=:default)
```

original point A | length | width | color | z location | use default setting of triangle

300, 200
x coordinate, y coordinate of A



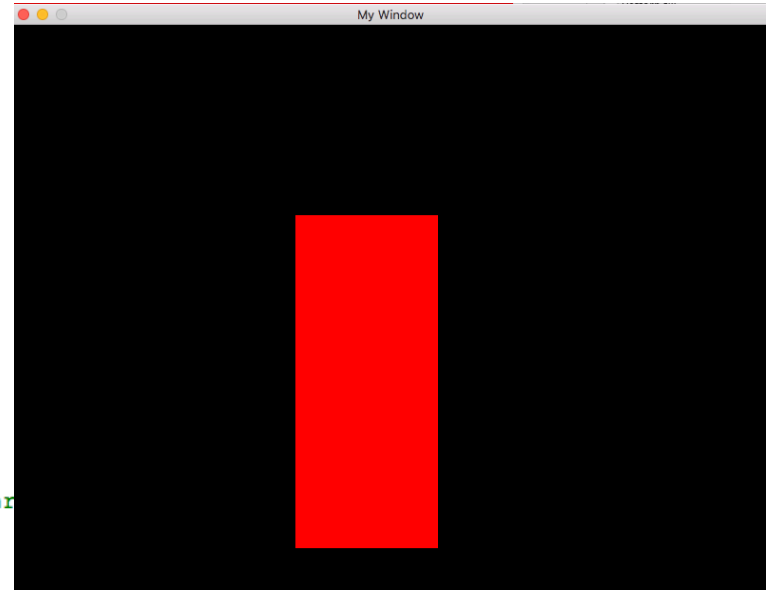
Demo > draw_rect

```
Gosu.draw_rect(300, 200, 150, 350, Gosu::Color::RED, 2, mode=:default)
```

Your code stored in **mywindow.rb**

```
1  # load gosu library
2  require 'gosu'
3
4  # define the "prototype" of your application
5  class MyWindow < Gosu::Window
6
7      # initial values when application load
8      def initialize()
9          super 800,600,false
10         self.caption = "My Window"
11     end
12
13     # put main logic of program inside update()
14     def update()
15     end
16
17     # callback to handle inputs from user, such as mouse and keyboar
18     def button_down(id)
19     end
20
21     # draw the interface inside draw
22     def draw()
23         Gosu.draw_rect(300, 200, 150, 350, Gosu::Color::RED, 2, mode=:default)
24     end
25 end
26
27 # start your application by constructing an instance of prototype
28 window = MyWindow.new()
29 window.show()
```

What you see after running it in Terminal



put your code for drawing here in the template