John Romero Programming Proverbs

- 3. “Keep your code absolutely simple. Keep looking at your functions and figure out how you can simplify further.”

- John Romero, “The Early Days of Id Software - John Romero @ WeAreDevelopers Conference 2017”

Overview of TouchGUI

- documentation ⟨`http://floppsie.comp.glam.ac.uk/touchgui/homepage.html`⟩
- you can obtain a copy of the source code for `touchgui` by:

-
  ```
  $ cd
  $ mkdir -p Sandpit
  $ cd Sandpit
  $ git clone https://github.com/gaiusm/touchgui
  ```

Overview of TouchGUI

- `touchgui` is a simple tablet based gui for Python/Pygame
    - it allows tiles to be created from images, colours or glyphs
    - each tile has a number of callbacks which are called whenever a tap or double tap occurs

Overview of TouchGUI

- a tile maybe in one of the following four states: images for the tile when in the frozen, active, activated or pressed state
  - the frozen state is when the tile cannot be pressed
    - (the application might choose to disable the tile)

- the active state is when the tile can be pressed by the user
  - the activated state is when the mouse pointer is hovering over the tile (but not pressed)
  - finally the pressed state is when the button is tapped.

Touchgui in the labs

- `touchgui` is installed in the J109 labs
  - you need to explicitly alter the `PYTHONPATH`

- you can do this on the command line and run your `touchmap` by:
  - (this assumes you have already downloaded and configured touchmap from previous weeks)

Touchgui in the labs

-
  ```
  $ cd
  $ cd Sandpit/build-touchmap
  $ PYTHONPATH=.:../touchmap-0.1:../touchgui python ../touchmap-0.1/touchmap.py
  ```

- the `PYTHONPATH` environment variable is set to search the current directory (the first `.`)
  - then search `../touchmap-0.1` and lastly search `../touchgui` for any python modules (before searching the system installed libraries)
  - note the path separator `:`

- after setting the `PYTHONPATH` the `python` interpreter is executed which inherits this `PYTHONPATH` and starts interpreting `../touchmap-0.1/touchmap.py`

Touchgui in the labs

- using a suitable file manager examine the contents of `touchgui`
- in particular examine the library of creative common images
- maybe make a note of icons you might find useful for your touchmap implementation

Single glyph button example using touchgui

```
!/usr/bin/env python

import pygame, touchgui, touchguipalate, touchguiconf, math, os
from pygame.locals import *

# display_width, display_height = 1920, 1080
display_width, display_height = 800, 600
display_width, display_height = 1920, 1080
full_screen = False
full_screen = True
toggle_delay = 250
```

Single glyph button example using touchgui

```
def event_test (event):
    if (event.type == KEYDOWN) and (event.key == K_ESCAPE):
        myquit (None)


def myquit (name = None, tap = 1):
    print "quit called"
    pygame.display.update ()   # need this to see the button pressed before we quit
    pygame.time.delay (toggle_delay * 2) #  delay program so we see the button change
    pygame.quit ()   #  now shutdown pygame
    quit ()   #  and shutdown python


def myreturn (name, tap):
    print "return called"
```

Single glyph button example using touchgui

- the function `myquit` is a callback which is called when the off button is pressed
  - both parameters are optional
  - a single parameter is allowed and then `tap` will be assigned to `1`

Single glyph button example using touchgui

-
  ```
  def imagedir (name):
      return os.path.join (touchguiconf.touchguidir, name)

  def button_list (name):
      return [touchgui.image_gui (imagedir ("images/PNG/White/2x/%s.png") \
                   % (name)).white2grey (.5),
              touchgui.image_gui (imagedir ("images/PNG/White/2x/%s.png") \
                   % (name)).white2grey (.1),
              touchgui.image_gui (imagedir ("images/PNG/White/2x/%s.png") \
                   % (name)),
              touchgui.image_gui (imagedir ("images/PNG/White/2x/%s.png") \
                   % (name)).white2rgb (.1, .2, .4)]
  ```

- note the \ is a line continuation character

Single glyph button example using touchgui

- `button_list` is a function which returns a list of four images
- the four images in order represent the four states
  - frozen, active, activated or pressed state
- `button_list` takes a single white image and produces four images
  - darkgrey using `white2grey (.5)` representing frozen
  - lightgrey using `white2grey (.1)` representing active
  - brillant white representing activated
  - dark blue `white2rgb (.1, .2, .4)` representing pressed

Single glyph button example using touchgui

```
#
#  buttons - create two buttons and return them as a list.
#

def buttons ():
    return [touchgui.image_tile (button_list ("power"),
                                 touchgui.posX (0.95), touchgui.posY (1.0),
                                 100, 100, myquit),
            touchgui.image_tile (button_list ("return"),
                                 touchgui.posX (0.0), touchgui.posY (1.0),
                                 100, 100, myreturn)]
```

Single glyph button example using touchgui

```
def main ():
    pygame.init ()
    if full_screen:
        gameDisplay = pygame.display.set_mode ((display_width, display_height), \
                                               FULLSCREEN)
    else:
        gameDisplay = pygame.display.set_mode ((display_width, display_height))

    pygame.display.set_caption ("Simple Test")
    touchgui.set_display (gameDisplay, display_width, display_height)

    forms = buttons ()
    gameDisplay.fill (touchguipalate.black)
    touchgui.select (forms, event_test)

main ()
```

Single glyph button example using touchgui

- `touchgui.select` can take 2 parameters (it can also take more - in future weeks this will be covered)
- the second parameter allows you to test `pygame` events
- the first parameter is a list of buttons on the touch device
  - only buttons in this list can be activated (mouse over) and/or tapped

Single glyph button example using touchgui

- `touchgui.image_tile` takes 6 parameters
  - `button_list` is the list of the four state images
  - `touchgui.posX` and `touchgui.posY` converts a floating point value in the range `0.0..1.0` onto the X or Y resolution of the screen (or window)
  - parameters 4 and 5 are the x and y image size
  - parameter 6 is the call back if tapped or double tapped