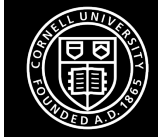
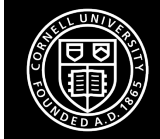


ECE 5725
Embedded Operating Systems
Lecture 13

Prof. Joseph F. Skovira



A few items

- Homework 2
- Lab Report 1
- Lab 2, Week 2
- Break!
- Green Hills

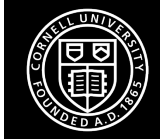
Scan to register

Software That Never Fails and Cannot Be Hacked:

Because our lives depend on it

Wednesday, 9/29 ◆ 7:30 pm ET ◆ Location: 206 Upson hall

Dan O'Dowd, CEO of Green Hills Software and designer of the most secure OS in the world, talks about preventing Cyber Armageddon with software that never fails and can't be hacked.



A few more Lab 2 items….

Bounce is too fast

Ball off screen or stuck at edge…

Balls stick together on collision

Bounce pattern repeats

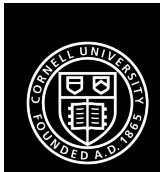
No Touch Bounce

Black on Black

Python, by example

t4. py # lists

t5.py
t7.py # dictionaries



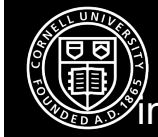
PyGame input

First: Buster to Wheezy Downgrade…..

Draw some ‘buttons’

Detect when the screen ‘buttons’ are pressed

Read input from Touch Screen



PyGame Input


```
import pygame
from pygame.locals import *   # for event MOUSE variables
import os

os.putenv('SDL_VIDEODRIVER', 'fbcon')   # Display on piTFT
os.putenv('SDL_FBDEV', '/dev/fb1')
os.putenv('SDL_MOUSEDRV', 'TSLIB')     # Track mouse clicks on piTFT
os.putenv('SDL_MOUSEDEV', '/dev/input/touchscreen')

pygame.init()
pygame.mouse.set_visible(False)
WHITE = 255, 255, 255
BLACK = 0,0,0
screen = pygame.display.set_mode((320, 240))

my_font = pygame.font.Font(None, 50)
my_buttons = { 'button1':(80,180), 'button2':(240,180)}
screen.fill(BLACK)              # Erase the Work space

for my_text, text_pos in my_buttons.items():
   text_surface = my_font.render(my_text, True, WHITE)
   rect = text_surface.get_rect(center=text_pos)
   screen.blit(text_surface, rect)

pygame.display.flip()
```

PyGame Input

```
while True:
   for event in pygame.event.get():
      if(event.type is MOUSEBUTTONDOWN):
         pos = pygame.mouse.get_pos()
      elif(event.type is MOUSEBUTTONUP):
         pos = pygame.mouse.get_pos()
         x,y = pos

         if y > 120:
            if x < 160:
               print ”button1 pressed"
            else:
               print ”button2 pressed
```

Pygame: useful commands

```
pygame.init()   # initialize pygame library

my_screen=pygame.display.set_mode(size)      # setup drawing surface

my_image = pygame.image.load(ball.png)      # load an image = surface

my_image_rect = my_image.get_rect()      # establishes rect for a surface

my_image_rect = my_image_rect.move(1,1)      # metadata: rect position
```

Pygame: useful commands

```
my_screen.fill(0,0,0)   # initialize drawing surface color

my_screen.blit(my_image, my_image_rect)      # combine surfaces

pygame.display.flip      # display working screen surface


my_image_rect = my_image_rect.inflate(-20, -20)   # grow/shrink rect

pygame.display.update (draw_rects)      # display rects on screen

pygame.event.get     # get events from the queue
```