```
import pygame, sys
from pygame.locals import *

WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 0, 128)
YELLOW = (153, 153, 0)
RED = (255, 0, 0)

class Button():

    SQUARE = 1
    CIRCLE = 2

    def __init__(self, position, text, shape, surface, check=False, color=BLACK,
                 size=50, text_size=16):
        #Initialize main variables
        self.position = position
        self.color = color
        self.size = size
        self.checked = check
        self.shape = shape
        self.surf = surface

        #Create text and text surface
        button_font = pygame.font.Font('freesansbold.ttf', text_size)
        self.text_surface = button_font.render(text, True, self.color)
        text_rect = self.text_surface.get_rect()

        #Create button surface and draw button
        self.button_surface = pygame.Surface((size, size))
        self.draw_button(self.button_surface, self.color,
                         self.button_surface.get_rect(), 3)
        self.checked_rect = self.button_surface.get_rect()
        self.checked_rect.inflate_ip(-10, -10)
        if(self.checked):
            self.draw_button(self.button_surface, self.color, self.checked_rect)

        #Create main checkbox surface
        #Both text surface and button surface will be blitted here
        surface_width = self.size + 5 + text_rect.width
        surface_height = max(self.size, text_rect.height)
        self.surface = pygame.Surface((surface_width, surface_height))
        self.rect = self.surface.get_rect()
        self.rect.topleft = self.position

        self.update_surface()

    def update_surface(self):
        self.surface.fill(WHITE)
        self.button_surface.fill(WHITE)
        #Redraw button rects
        self.draw_button(self.button_surface, self.color,
                         self.button_surface.get_rect(), 3)
        if(self.checked):
```

```
            self.draw_button(self.button_surface, self.color, self.checked_rect)
        self.surface.blit(self.button_surface, (0, 0))

        #do some math and redraw text
        rect = self.surface.get_rect()
        y = rect.centery
        y -= ((self.text_surface.get_rect()).height)/2
        self.surface.blit(self.text_surface, (self.size + 5, y))


    def get_surface(self):
        return self.surface

    def draw_surface(self):
        self.surf.blit(self.surface, self.position)

    def get_if_checked(self):
        return self.checked

    def set_if_checked(self, check):
        self.checked = check
        self.surface.fill(WHITE)
        self.button_surface.fill(WHITE)
        #Redraw button rects
        self.draw_button(self.button_surface, self.color,
                         self.button_surface.get_rect(), 3)
        if(check):
            self.draw_button(self.button_surface, self.color, self.checked_rect)
        self.surface.blit(self.button_surface, (0, 0))

        #do some math and redraw text
        rect = self.surface.get_rect()
        y = rect.centery
        y -= ((self.text_surface.get_rect()).height)/2
        self.surface.blit(self.text_surface, (self.size + 5, y))

class CheckBox(Button):

    def __init__(self, position, text, check=False, color=BLACK,
                 size=50, text_size=16):
        Button.__init__(self, position, text, Button.SQUARE, check=False,
                        color=BLACK, size=50, text_size=16)

    def draw_button(self, surface, color, rect, thickness=0):
        pygame.draw.rect(surface, color, rect, thickness)

    def click_action(self):
        self.checked = not self.checked
        self.update_surface()
        MAIN_SURF.blit(self.surface, self.position)

    def check_for_click(self):
        if self.rect.collidepoint(pygame.mouse.get_pos()):
            self.click_action()
            return True
        else:
```

```
            return False

    class RadioButton(Button):

        def __init__(self, position, text, shape, surface, check=False, color=BLACK,
                     size=50, text_size=16):
            #for individual radio buttons so they will inherit
            Button.__init__(self, position, text, shape, surface, check, color, size,
                text_size)

        def draw_button(self, surface, color, rect, thickness=0):
            pygame.draw.ellipse(surface, color, rect, thickness)

    class RadioButtonGroup():

        def __init__(self, (x, y), button_number, surface, check=False, color=BLACK,
                     size=50, text_size=16, names=[]):
            self.surf = surface
            self.total_buttons = button_number
            self.buttons = []
            for button in range(self.total_buttons):
                if (len(names) == 0):
                    name = ""
                else:
                    name = names[button]
                self.buttons.append(RadioButton((x, y+((y + 10) * button)), name,
                    Button.CIRCLE, self.surf,
                                                    size=50, text_size=16))

            self.buttons[check].set_if_checked(True)

        def check_for_click(self):
            clicked = False
            for counter in range(len(self.buttons)):
                button = self.buttons[counter]
                if (button.rect.collidepoint(pygame.mouse.get_pos())) and (not button
                    .checked):
                    button.set_if_checked(True)
                    clicked = True
                    break
            if(clicked):
                for new_counter in range(len(self.buttons)):
                    if(new_counter != counter):
                        self.buttons[new_counter].set_if_checked(False)

        def click_action(self, button_number):
            if(not self.checked):
                self.set_if_checked(True)
            self.checked = not self.checked
            self.update_surface()
            self.surf.blit(self.surface, self.position)

        def draw_surface(self):
            for button in self.buttons:
                button.draw_surface()
```

```
        def get_selected(self):
            for button in range(self.total_buttons):
                if(self.buttons[button].get_if_checked()):
                    return button
```