```
  1: # Example code relating to interactive storyboarding
  2: # By Brygg Ullmer, Clemson University
  3: # Begun 2023-11-09
  4:
  5: from tkinter import *
  6: import PIL.Image, PIL.ImageTk #image manipulation package
  7:
  8: #WIDTH=1024
  9:
 10: screenStates = ['unsdg2', 'unsdg4']
 11: imgAddUser   = 'person-add-iconic1'
 12:
 13: #actorNames          = {a1: "John", a2: "Jane", s1: "screen", b1: "addUser"}
 14: actorOriginalPos     = {}
 15: selectedActor        = None
 16: selectedActorName    = None
 17: selectedActorOrigPos = None
 18:
 19: def helloCB():
 20:   print("hello was pushed")
 21:
 22: ######################## build UI ######################
 23:
 24: imP1 = imTk1 = None
 25: imP2 = imTk2 = None
 26:
 27: #https://stackoverflow.com/questions/51591456/can-i-use-rgb-in-tkinter
 28: #translates rgb values of type int to a tkinter friendly color code
 29: def rgb2tk(r, g, b):
 30:   return "#%02x%02x%02x" % (r,g,b)
 31:
 32: c                    = None #canvas handle, sigh; should be moved into a class
 33: selectedCanvasObject = None #ID (1,2,3...) of a selected object within canvas c
 34: lastDragXY           = None #coordinates of where a mouse-drag sequence began
 35: img1                 = None
 36:
 37: ######################## build user interface ######################
 38:
 39: def buildUI(f1Screens, f2Spatial, f3Controls):
 40:   global imP1, imTk1, imP2, imTk2, c, img1
 41:
 42:   imgAddUserFn   = 'person-add-iconic1.png'
 43:   imP1  = PIL.Image.open(imgAddUserFn)
 44:   imTk1 = PIL.ImageTk.PhotoImage(imP1)
 45:
 46:   #b = Button(f3Controls, text="add actor", command=helloCB) # Create a label with
words
 47:   b  = Button(f3Controls, image=imTk1, command=addCanvasItem)
 48:   b.pack(side=LEFT, expand=True, fill=BOTH)
 49:
 50:   screenFilenames = ['unsdg2.png', 'unsdg4.png']
 51:   imP2   = PIL.Image.open(screenFilenames[0])
 52:   imTk2  = PIL.ImageTk.PhotoImage(imP2)
 53:   label1 = Label(f1Screens, image=imTk2)
 54:   label1.pack()
 55:
 56:   bgColor = rgb2tk(10, 10, 10)
 57:   #c = Canvas(f2Spatial, bg="orange", height=200, width=1024)
 58:   c = Canvas(f2Spatial,  bg=bgColor,  height=400, width=1024)
 59:   c.pack()
 60:
 61:   img1Fn = 'clemson12d2.png'
 62:   img1   = PhotoImage(file=img1Fn) #transparent image
 63:   c.create_image(100, 100, image=img1,anchor='ne')
 64:
 65:   r1Coords = (10, 10, 60, 60)
 66:   r1 = c.create_rectangle(r1Coords, fill="white")
 67:
```

```
   68:   r2Coords = (70, 10, 120, 60)
   69:   r2 = c.create_rectangle(r2Coords, fill="orange")
   70:
   71:   #print("canvas rectangle ids:", r1, r2)
   72:   #c.move(r1, 50, 50)
   73:
   74:   c.bind("<Button-1>",  on_click)
   75:   c.bind("<B1-Motion>", on_drag)
   76:
   77: ################### mouse click callback ##################
   78:
   79: #next pulls from https://stackoverflow.com/questions/65189412/python-canvas-move-i
tems-with-mouse-tkinter
   80:
   81: def on_click(event):
   82:   global c, selectedCanvasObj, lastDragXY
   83:
   84:   x, y = event.x, event.y
   85:   csr  = 10 #click search radius
   86:   x1, y1, x2, y2 = x-csr, y-csr, x+csr, y+csr
   87:   #print("click event:", event)
   88:
   89:   selected = c.find_overlapping(x1, y1, x2, y2)
   90:   if selected: selectedCanvasObj = selected[-1]
   91:   else:        selectedCanvasObj = None
   92:
   93:   lastDragXY = (x,y) #for calculating dx, dy movement changes with drag
   94:
   95:   #print("selected:", selectedCanvasObj)
   96:
   97: ################### mouse drag callback ##################
   98:
   99: def on_drag(event):
  100:   global c, selectedCanvasObj, lastDragXY
  101:   #print("drag event:", event)
  102:
  103:   x0, y0 = lastDragXY
  104:   x1, y1 = event.x, event.y
  105:   dx, dy = x1-x0, y1-y0
  106:   lastDragXY = (x1,y1)
  107:
  108:   c.move(selectedCanvasObj, dx, dy)
  109:   print(">>", selectedCanvasObj, x1, y1)
  110:
  111: ################### add Canvas Item ##################
  112:
  113: def addCanvasItem():
  114:   global c
  115:   rCoords = (100, 100, 150, 150)
  116:   r = c.create_rectangle(rCoords, fill="purple")
  117:
  118: ######################## main ######################
  119:
  120: root = Tk() # Create the root (base) window
  121:
  122: f1Screens  = Frame(root)
  123: f2Spatial  = Frame(root)
  124: f3Controls = Frame(root)
  125: buildUI(f1Screens, f2Spatial, f3Controls)
  126:
  127: for frame in [f1Screens, f2Spatial, f3Controls]:
  128:   frame.pack(side=TOP, expand=True, fill=BOTH)
  129:
  130: root.mainloop() # Start the event loop
  131:
  132: ### end ###
```