```
102:   ######################## draw ########################
104:   def draw(self):
105:     for actor in self.actors: actor.draw()
107:   ######################## draw ########################
109:   def onMouseDown(self, pos):
110:     for actor in self.actors:
111:       if actor.collidepoint(pos):
112:         category = self.actor2category[actor]
113:         print("pushed:", category)
114:         self.animateSelected(category)
116: ################################################
118: enoUiH = enoUiHomelessnessPgz()
120: ######################## draw ########################
121: def draw():
122:   screen.clear()
123:   enoUiH.draw()
124:
125: ######################## on_mouse_down ########################
126: def on_mouse_down(pos):
127:   enoUiH.onMouseDown(pos)
```

pySdgEx01g2.py

```
 5: import tkinter as tk
 6: from enoIgridTk import *
 7:
 8: top  = tk.Tk()
 9: tkig = enoIgridTk(top, numButtons=17, imageLabelDir="sdg", useImageLabels=True)
10:
11: sdgIdx = tk.IntVar()
12:
13: def sdgSliderCb(event):
14:   global tkig, sdgIdx
15:   tkig.buttonCallback(sdgIdx.get())
16:
17: slider = tk.Scale(top, var=sdgIdx, from_=1, to=18, command=sdgSliderCb, orient=tk.HORIZONTAL)
18: slider.pack(expand=1, fill=tk.X)
19:
20: top.mainloop()
```

enoIgridTk2.py

2

```
10: class enoIgridTk:

27:   def __init__(self, tkParent, **kwargs):
31:     self.buildGui(tkParent)

72:   def buildGui(self, tkParent):
73:     self.igridParent = tkParent
74:     self.igridFrame  = tk.Frame(tkParent)
75:     self.igridFrame.pack(expand=1)
76:
77:     rowFrame = tk.Frame(self.igridFrame) # invisible bundle of UI widgets
78:     rowFrame.pack(expand=1)
79:     colNum   = 1
80:     self.buttonMapIdx = {}
81:
82:     if self.useImageLabels:
83:       self.imageMapNorm = {}
84:       self.imageMapDs   = {}
85:
86:     for i in range(self.numButtons):
87:       cb = partial(self.callbackFunc, i+1)
88:
89:       if self.useImageLabels:
90:         imgNFn = self.genImageNormFn(i+1); imgDFn = self.genImageDsFn(i+1)
91:         imgN  = tk.PhotoImage(file=imgNFn); self.imageMapNorm[i] = imgN
92:         imgD  = tk.PhotoImage(file=imgDFn); self.imageMapDs[i]   = imgD
93:         b1    = tk.Button(rowFrame, image=imgN, command=cb)
94:       else:
95:         buttonLabel = "B%i" % (i+1)
96:         b1 = tk.Button(rowFrame, text=buttonLabel, command=cb, width=self.buttonWidth)
97:
98:       self.buttonMapIdx[i] = b1
99:
100:      b1.pack(side=tk.LEFT)
101:      colNum += 1
102:
103:      if colNum  > self.numPerRow:
104:        rowFrame = tk.Frame(self.igridFrame);
105:        rowFrame.pack(expand=1, side=tk.TOP)
106:        colNum = 1
107:
108:      rowFrame.pack()
```

```
 1: # Brygg Ullmer, Clemson University
 2: # Begun 2022-11-01
 3: # Content engaging https://github.com/DataKind-DC/homelessness-service-navigator
 4:
 5: import yaml
 6:
 7: class enoDomHomelessness:
 8:
 9:   yamlFn = "homelessness/dkdc01.yaml"
10:
11:   yamlD        = None
12:   imagePath    = None
13:   yOffset      = None #placeholder assignments until read in or assigned
14:   xOffset      = None
15:   categories   = None
16:   descriptions = None
17:
18:   ######################## constructor ######################
19:
20:   def __init__(self):
21:     self.readYaml()
22:
23:   ######################## read YAML ########################
24:
25:   def readYaml(self):
26:     try:
27:       yf             = open(self.yamlFn)  #open yamlFn filename for reading
28:       yd = self.yamlD = yaml.safe_load(yf) #load and parse YAML content
29:
30:       self.xOffset    = yd['positions']['xOffset']
31:       self.yOffset    = yd['positions']['yOffset']
32:       self.imagePath  = yd['paths']['images']
33:
34:       self.categories   = yd['categories']
35:       self.descriptions = yd['descriptions']
36:
37:     except: print("Problem in enoDomHomelessness:readYaml")
38:
39:   ######################## get categories ######################
40:
41:   def getCategories(self): return self.categories
42:
43:   ######################## get Image Filename ######################
44:
45:   def getImageFn(self, category):
46:     descr = self.getDescr(category)
47:     iconN = descr['icon']
48:     result = "%s/%s" % (self.imagePath, iconN)
49:     return result
```



```
 1: # Brygg Ullmer, Clemson University
 2: # Begun 2022-11-01
 3: # Content engaging https://github.com/DataKind-DC/homelessness-service-navigator
 4:
 5: paths:
 6:   origSrc: https://github.com/DataKind-DC/homelessness-service-navigator
 7:   images:  dkdc
 8:
 9: positions:
10:   yOffset: 110 # y distance between icons
11:   xOffset: 350 # x distance toward icon horizontal shift
12:
13: categories: [health, food, housing, employment, transit, goods]
14:
15: descriptions:
16:   health:     {icon: dkdc_health1,     visuals: []}
17:   food:       {icon: dkdc_food1,       visuals: []}
18:   housing:    {icon: dkdc_housing1,    visuals: []}
19:   employment: {icon: dkdc_employment1, visuals: []}
20:   transit:    {icon: dkdc_transit1,    visuals: []}
21:   goods:      {icon: dkdc_goods1,      visuals: []}
22:
23: ### end ###
24:
```