DetectWink.py

```python
import numpy as np
import cv2
import os
from os import listdir
from os.path import isfile, join
import sys

def detectWink(frame, location, ROI, cascade):
    eyes = cascade.detectMultiScale(
        ROI, 1.15, 3, 0|cv2.CASCADE_SCALE_IMAGE, (10, 20))
    for e in eyes:
        e[0] += location[0]
        e[1] += location[1]
        x, y, w, h = e[0], e[1], e[2], e[3]

        cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 0, 255), 2)
    return len(eyes) == 1    # number of eyes is one

def detect(frame, faceCascade, eyesCascade):
    gray_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # possible frame pre-processing:
    # gray_frame = cv2.equalizeHist(gray_frame)
    # gray_frame = cv2.medianBlur(gray_frame, 5)

    scaleFactor = 1.15 # range is from 1 to ..
    minNeighbors = 3   # range is from 0 to ..
    flag = 0|cv2.CASCADE_SCALE_IMAGE # either 0 or 0|cv2.CASCADE_SCALE_IMAGE
    minSize = (30,30) # range is from (0,0) to ..
    faces = faceCascade.detectMultiScale(
        gray_frame,
        scaleFactor,
        minNeighbors,
        flag,
        minSize)

    detected = 0
    for f in faces:
        x, y, w, h = f[0], f[1], f[2], f[3]
        faceROI = gray_frame[y:y+h, x:x+w]
        if detectWink(frame, (x, y), faceROI, eyesCascade):
            detected += 1
            cv2.rectangle(frame, (x,y), (x+w,y+h), (255, 0, 0), 2)
        else:
            cv2.rectangle(frame, (x,y), (x+w,y+h), (0, 255, 0), 2)
    return detected


def run_on_folder(cascade1, cascade2, folder):
    if(folder[-1] != "/"):
        folder = folder + "/"
    files = [join(folder,f) for f in listdir(folder) if isfile(join(folder,f))]
```

```
53
54      windowName = None
55      totalCount = 0
56      for f in files:
57          img = cv2.imread(f, 1)
58          if type(img) is np.ndarray:
59              lCnt = detect(img, cascade1, cascade2)
60              totalCount += lCnt
61              if windowName != None:
62                  cv2.destroyWindow(windowName)
63              windowName = f
64              cv2.namedWindow(windowName, cv2.WINDOW_AUTOSIZE)
65              cv2.imshow(windowName, img)
66              cv2.waitKey(0)
67      return totalCount
68
69  def runonVideo(face_cascade, eyes_cascade):
70      videocapture = cv2.VideoCapture(0)
71      if not videocapture.isOpened():
72          print("Can't open default video camera!")
73          exit()
74
75      windowName = "Live Video"
76      showlive = True
77      while(showlive):
78          ret, frame = videocapture.read()
79
80          if not ret:
81              print("Can't capture frame")
82              exit()
83
84          detect(frame, face_cascade, eyes_cascade)
85          cv2.imshow(windowName, frame)
86          if cv2.waitKey(30) >= 0:
87              showlive = False
88
89      # outside the while loop
90      videocapture.release()
91      cv2.destroyAllWindows()
92
93
94  if __name__ == "__main__":
95      # check command line arguments: nothing or a folderpath
96      if len(sys.argv) != 1 and len(sys.argv) != 2:
97          print(sys.argv[0] + ": got " + len(sys.argv) - 1
98                + "arguments. Expecting 0 or 1:[image-folder]")
99          exit()
100
101     # load pretrained cascades
102     face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades
103                                          + 'haarcascade_frontalface_default.xml')
104     eye_cascade = cv2.CascadeClassifier(cv2.data.haarcascades
105                                         + 'haarcascade_eye.xml')
```

```
106
107     if(len(sys.argv) == 2): # one argument
108         folderName = sys.argv[1]
109         detections = run_on_folder(face_cascade, eye_cascade, folderName)
110         print("Total␣of␣", detections, "detections")
111     else: # no arguments
112         runonVideo(face_cascade, eye_cascade)
```