*Face Analyzer  
Recognizing Eyes Color using OpenCV and Webcolors*

Table of Contents

[Overview 2](#_Toc80196961)

[Scope 2](#_Toc80196962)

[Pre-requisites 2](#_Toc80196963)

[Setup 3](#_Toc80196964)

[Final Words 10](#_Toc80196965)

Overview

It is said that the eyes are the doorway to the heart; our eyes are a reflection of our inner self and emotions. But what gives eyes their color?  
The iris color is determined by the amount of melanin pigmentation. The more pigment there is, the darker the iris will be. Blue, gray and green eyes are lighter because there is less melanin inside the iris.

Scope

The target of this tutorial is to develop a lightweight command line based utility, through Python based modules to automatically detect faces in a static image and to recognize the color of the eyes of the spotted faces.  
  
If this tutorial intrigues you, then grab its code from the following GitHub repository: “<https://github.com/bassemmarji/FaceAnalyzer/> ”.

# Pre-requisites

The following components come into play:

* **OpenCV**: is an open-source library for computer vision, machine learning and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java and it is used for all sorts of image and video analysis like facial detection and recognition, photo editing, optical character recognition and a whole heap more.
* **Dlib**: is principally a C++ library however you can use a number of its functions from Python applications.
* **filetype**: is a small and dependency free Python package to infer file and MIME types.
* **imutils**: encompasses a series of functions to make basic image processing functions such as translation, rotation, resizing, skeletonization, displaying Matplotlib images, sorting contours, detecting edges.
* webcolors: is a library for working with color names and color values formats defined by HTML and CSS.

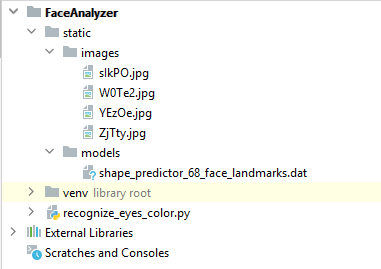
# Setup

To setup the environment, you need python3 installed on your system. It is highly recommended to setup a virtual environment which will host the needed libraries.

1. Create a virtual environment and activate it.
2. Create a file named requirements.txt and add the following lines to it.

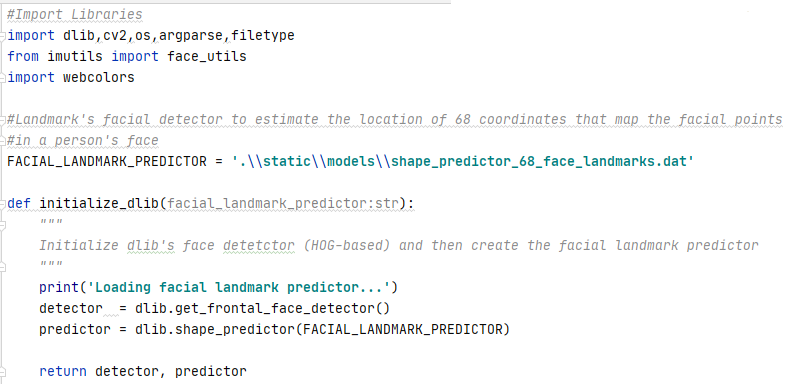
|  |
| --- |
| requirements.txt |
| opencv-python==4.4.0.46 dlib==19.17.0 imutils== 0.5.3  filetype ==1.0.7  webcolors==1.11.1 |

1. Now, let’s install the required libraries to the project.  
   pip install –r requirements.txt
2. Create a folder for our project called “FaceAnalyzer”.

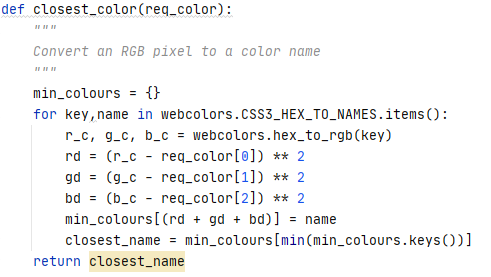
At the end, our folder structure will look like the following:  
  


**NB:**   
  
For the purpose of this article, we will use the “shape\_predictor\_68\_face\_landmarks.dat”;   
A dlib pre-trained model to estimate the location of 68 coordinates (x,y) that map the facial points on a person’s face. The model details are available in the below link   
<https://github.com/davisking/dlib-models>.

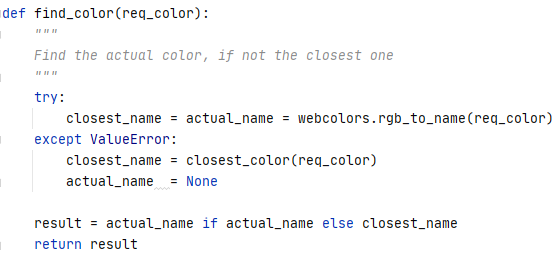
Let’s move into coding:  
 *#recognize\_eyes\_color.py*



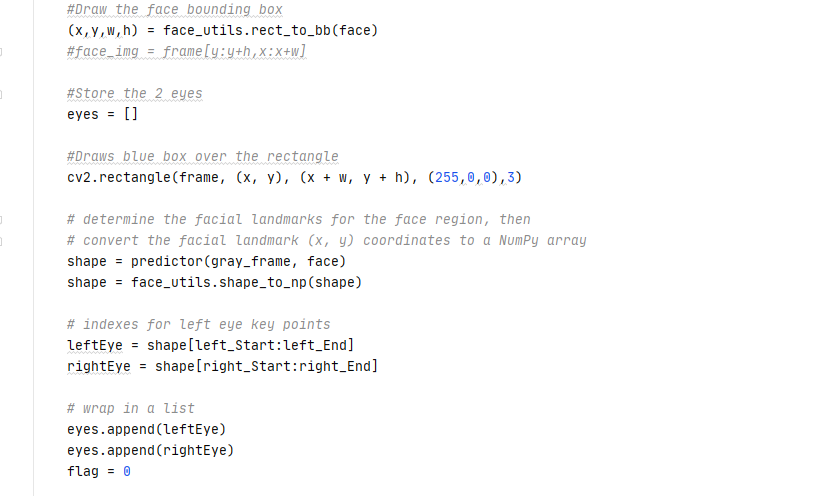
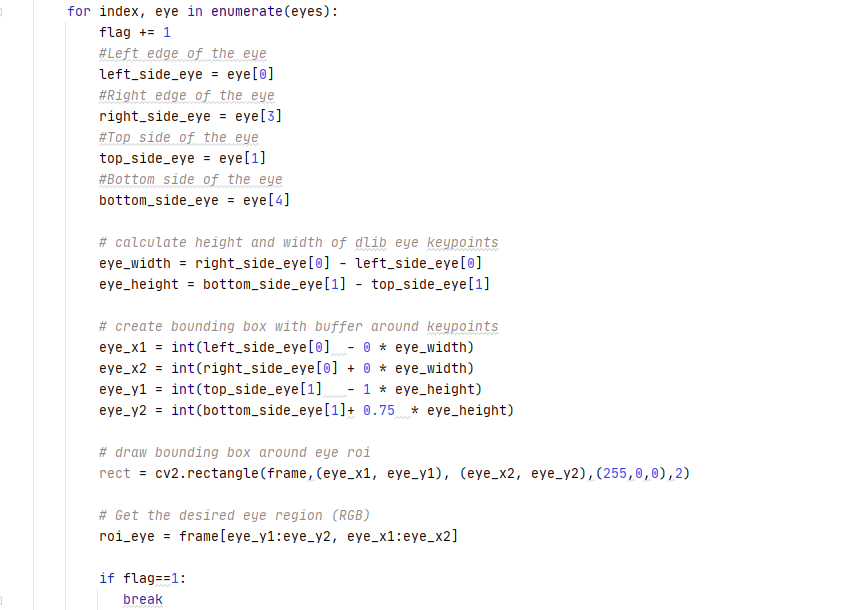
* This function initializes the dlib using the pre-trained model and returns:
  + detector: used for detecting the face in an image.
  + predictor: shape or landmark predictor used to predict the coordinates of a given shape. The facial landmark predictor is used to localize individual facial structures.

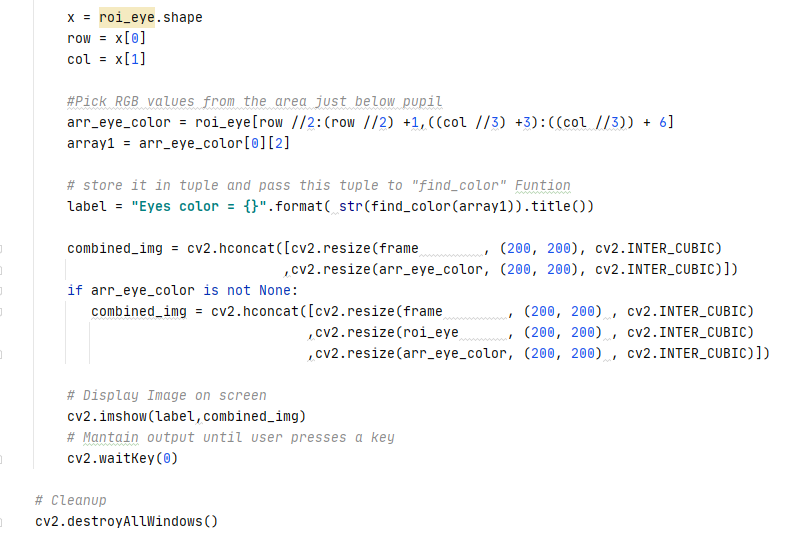


* This function maps an RGB values to the nearest color name it can find. It computes the sum of the difference between the red, green and blue numbers and chooses the smallest one.



* This function maps an RGB values to its actual corresponding color and if not found get the closest color.

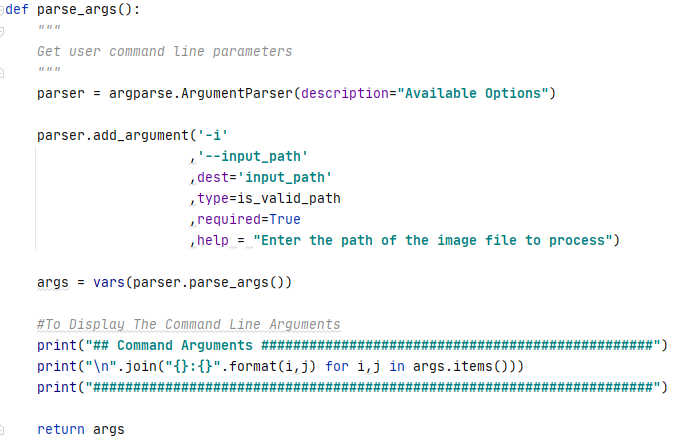
  
  




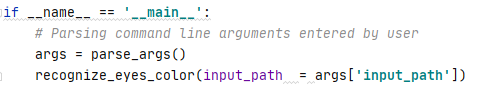
* This function constitutes the core of our program and performs the following:
  + Initialize the dlib face detector.
  + Read the input image.
  + Preserve a copy of the original image.
  + Convert the image into gray scale.
  + Detect faces in the grayed image.
  + Determine the landmarks of the left and right eyes.
  + Iterate throughout the faces detected and displays the positions of the faces detected.
  + Draw a bounding box around the faces detected.
  + Determine the facial landmarks for the face region then convert its coordinates to a Numpy array.
  + Draw a bounding box around the eye considered as the region of interest (ROI).
  + Pick the RGB value of the region situated directly below the pupil.
  + Display a window containing the original image, the eye designated as a region of interest and the color discerned.



* This function validates a path inputted as a parameter and ensures that it is a file path also it ascertains that the type of the file chosen is an image.



* This function defines and sets the appropriate constraints for the command line arguments to be specified by the user when running this utility.
  + input\_path: A required parameter containing the path of the image file to process associated with the predefined function “is\_valid\_path”.

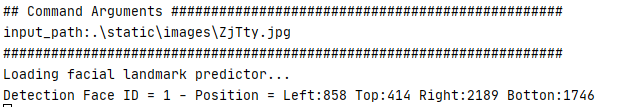


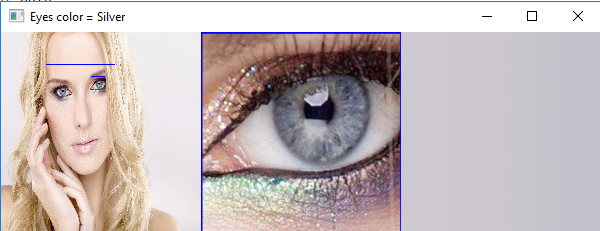
* The above represents the main function of our program.

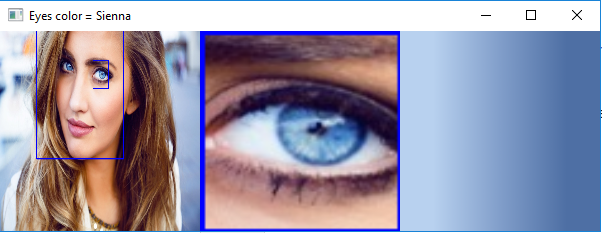
Let’s test our program:  
  
Kindly proceed as per the following steps:

***NB: Please make sure to select images with high resolution while testing this utility.***

Open up a terminal window and type the following in it:

**recognize\_eyes\_color –i ".\static\images\ZjTty.jpg"**  
The following summary will be displayed in the terminal:  
  




**recognize\_eyes\_color** -i ".\static\images\YEzOe.jpg"  
  


Final Words  
  
Several publications proclaim that eye color can indicate intelligence, personality traits and reflect the person’s health condition.  
Hope you found this article useful.

|  |  |
| --- | --- |
|  | ***Bassem Marji*** *is a project implementation manager at BLOM Bank with a proven track record of success.  He managed the implementation of over 50 projects and propelled the digital transformation of mission critical applications. He spends his free time discovering the latest technology trends in the IT field.* |