Image processing in Python by pillow

What is pillow?

- pillow is Python's image library based on PIL (Python Image Library)
- **PIL** is the **P**ython **I**maging **L**ibrary by Fredrik Lundh and Contributors
- pillow can be used for image processing like creating images, thumbnails, converting image format, rotating, resizing, image filters and ...
- pillow is a replacement for **PIL** for future usage since 2011

Why pillow?

- pillow supports a large number of image file formats including BMP, PNG, JPEG, and TIFF
- It needs very little code to make a desktop app
- It is lightweight image processing tools that is great in editing, creating & saving images.

1. How to install and use the pillow

In the pip write => **pip install pillow**

Using pillow => `import PIL.Image as MyImg`

2. Most common methods of pillow

- **Open and show the image**
- **Save image with different formats**
- **Get image Attributes like: Format, Size, Color Mode**
- **Make image thumbnail**
- **Resize image**

- **Rotate image with rotate method**
- **Crop image**
- **Rotate and flip image**
- **Add filters to image like Blur,Contour,Edge enhance ..**
- **Get image info**
- **Split image bands**
- **Adding watermark**
- **...**

3. Most common pillow methods

- **Open and show the image**

```
import PIL.Image as MyImg

#--------open & show ----
img: MyImg.Image =MyImg.open("me.png")
img.show()
#-----------------------
```

- **Save image with different format**

```
import PIL.Image as MyImg
#--------open  image ---------
img: MyImg.Image =MyImg.open("me.png")
#-------------save  image -----------------
#------- BMP, PNG, JPEG, TIFF ----------
img.save("me1.bmp"  ) #Simple save method
img.save("me2.jpg", "jpeg" ) #Save method with format
```

- **Get image Attributes like: Format, Size, Color Mode**

```
import PIL.Image as MyImg
#--------open  image ---------
img: MyImg.Image =MyImg.open("me.png")
#------------------------------
print("Type: " , type(img)) => 'PIL.PngImagePlugin.PngImageFile'
print("Size: " , img.size) =>(600,450)
print("Format: " , img.format) => PNG
print("Color Mode: " , img.mode) => => RGBA
#------------------------------
```

- Resize image

```
import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------resize and save  image -----------------

img_r=img.resize((200,200))
img_r.show()
img.save("me_resized.png" )
```

- Rotate image with rotate method

```
 import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------rotate and save image -----------------

img_rot=img.rotate(30)
img_rot.show()
img_rot.save("me_rotated.jpg" )
```

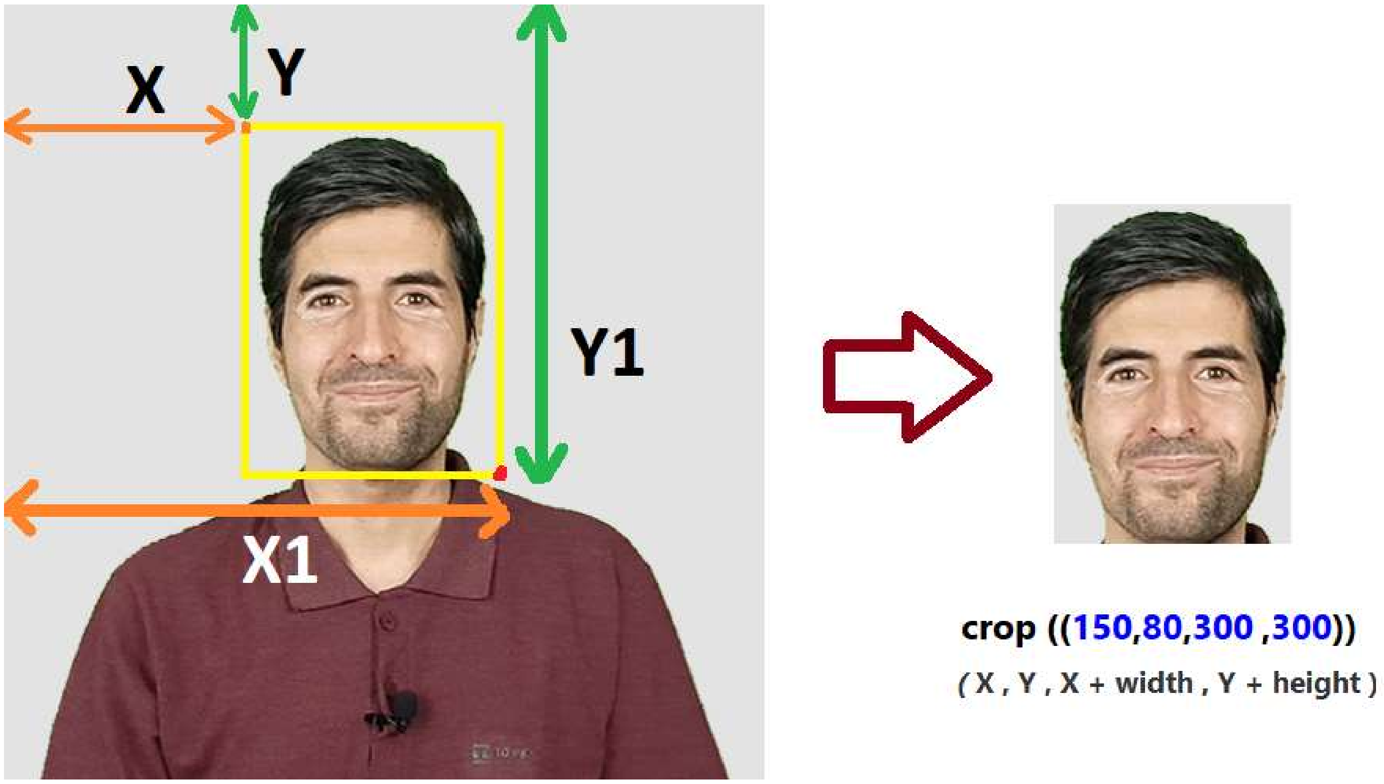
- Image Thumbnail

```
import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------build Thumbnail and save image -----------------

copy_img=img.copy()
copy_img.thumbnail((100,100) )

copy_img.show()
copy_img.save("me_thumb.png" )
```

- Crop image

```
import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------Crop and save image -----------------

# Crop box : 4 items tuple=>

# ( X , Y , X1 , Y1) => ( X , Y , X+width , Y+height )
img_cropped=img.crop((30,20,300 ,150))
```

```
img_cropped.show()
img_cropped.save("me_croped.png" )
```



```
crop ((150,80,300 ,300))
( X , Y , X + width , Y + height )
```

- **Rotate and Flip image**

```
import PIL.Image as MyImg

#--------open  image ---------
```

```
img: MyImg.Image =MyImg.open("me.png")
#-------------Rotate -Flip and save image -----------------

tr_image=img.transpose(MyImg.ROTATE_90)
tr_image=img.transpose(MyImg.ROTATE_180)
tr_image=img.transpose(MyImg.ROTATE_270)

#--------------------------

tr_image=img.transpose(MyImg.FLIP_LEFT_RIGHT)
tr_image=img.transpose(MyImg.FLIP_TOP_BOTTOM)
#--------------------------

tr_image.show()
tr_image.save("me_transpose.png" )
```

- Add filters to image like Blur, Contour, Edge enhance ...

```
import PIL.Image as MyImg

import PIL.ImageFilter as MyFilter

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")


#------------- Blur filter and save image -----------------
```

```
blur_image=img.filter(MyFilter.BLUR )

blur_image=img.filter(MyFilter.BoxBlur(3))
blur_image=img.filter(MyFilter.GaussianBlur(radius=2))

blur_image.show()
blur_image.save("me_filter_blur.png" )


#-------------  contour filter and save image --------------

contour_image=img.filter(MyFilter.CONTOUR)
contour_image.show()
contour_image.save( "me_filter_contour.png" )


#-------------  detail  filter and save image ---------------

detail_image=img.filter(MyFilter.DETAIL)
detail_image.show()
detail_image.save( "me_filter_detail.png" )


#---------  Edge_Enhance  filter and save image ------------

EdgeEn_image=img.filter(MyFilter.EDGE_ENHANCE)
EdgeEn_image.show()
EdgeEn_image.save( "me_filter_EdgeEn.png" )

#-----  Edge_Enhance_more  filter and save image --------

EdgeEnMo_image=img.filter(MyFilter.EDGE_ENHANCE_MORE)
EdgeEnMo_image.show()
EdgeEnMo_image.save( "me_filter_EdgeEnMo.png" )

#-----  Emboss filter and save image --------

Emboss_image=img.filter(MyFilter.EMBOSS)
Emboss_image.show()
Emboss_image.save( "me_filter_Emboss.png" )

#-----  Find_Edge filter and save image --------

FindEdge_image=img.filter(MyFilter.FIND_EDGES)
FindEdge_image.show()
FindEdge_image.save( "me_filter_FindEdge.png" )


#-----  Smooth filter and save image --------

Smooth_image=img.filter(MyFilter.SMOOTH)
Smooth_image.show()
Smooth_image.save( "me_filter_Smooth.png" )


#-----  Smooth_More filter and save image --------

SmoothMo_image=img.filter(MyFilter.SMOOTH_MORE)
SmoothMo_image.show()
SmoothMo_image.save( "me_filter_SmoothMo.png" )


#--------  Sharpen filter and save image -----------

Sharpen_image=img.filter(MyFilter.SHARPEN)
Sharpen_image.show()
Sharpen_image.save( "me_filter_Sharpen.png" )
```

- **Add simple text (watermark) to image**

```
import PIL.Image as MyImg

import PIL.ImageDraw as MyImgDraw

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------Add text to image -----------------
```

```
dr = MyImgDraw.Draw(img)
dr.text((28, 36), "Hello, Python!", fill=(255, 0, 0))

#-----------

img.show()
img.save("me_text_simple.png" )
```

- **Add text (Watermark) with custom font to image**

```
import PIL.Image as MyImg

import PIL.ImageDraw as MyImgDraw

import PIL.ImageFont as MyImgFont

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#--------- Add text with custom font to image ------- -----

MyFont = MyImgFont.truetype('Font/tahoma.ttf', 20)#otf, ttf

dr = MyImgDraw.Draw(img)
dr.text((0, 0), "Sample text", font=MyFont , fill =(255 ,0 , 0) )

#-----------

img.show()
img.save("me_text_font.png" )
```

- **Using image.info**

Returns a dictionary including data about the image.

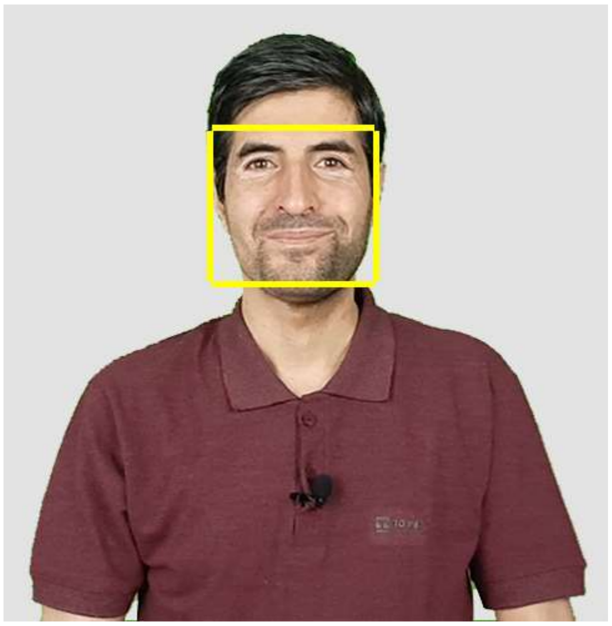
```
import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#-------------------------------

print(img.info)

#-------------------------------

for item in img.info :
    print(item)
#--------------------
```

- **Split and Merge image bands**

```
import PIL.Image as MyImg

#--------open  image ---------

img: MyImg.Image =MyImg.open("me.png")
#------------------------------

ColorBands=img.split()

# r, g, b = img.split()

#----------
r: MyImg.Image=ColorBands[0]
r.save("me.red.png")

#----------
g: MyImg.Image=ColorBands[1]
g.save("me.green.png")

#----------
b: MyImg.Image=ColorBands[2]
b.save("me.blue.png")

#--------------------------------

im=MyImg.merge("RGB", (r,g ,b ))
```

```
#------------------------------

img.show()
img.save("me_merged.png" )
```

Face recognition in Python by pillow



1. How to install & use the face detection in pillow

1. cmake => **pip install cmake**

# A software tool for managing the build process of software

2. dlib => **pip install dlib # machine learning library**

# optional => In the pip write => **pip install pillow**

3. face-recognition => **pip install face-recognition**

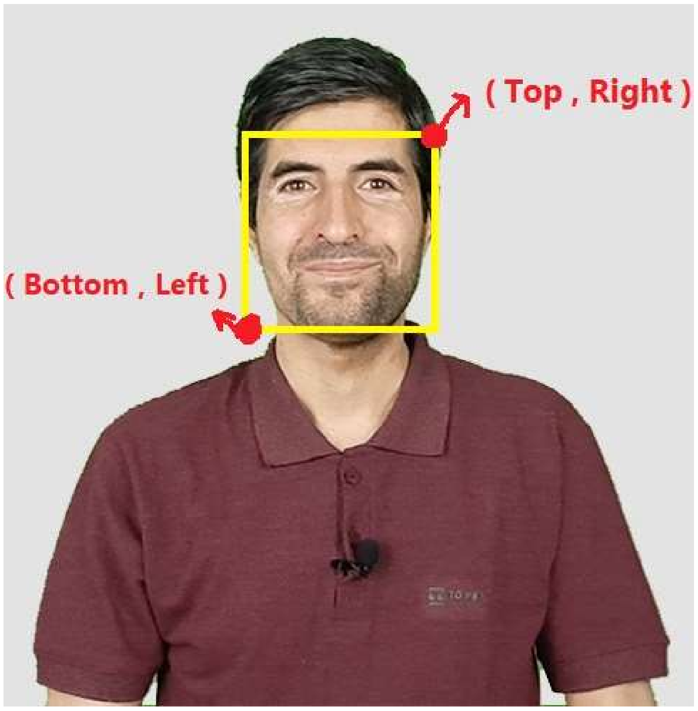
```
import face_recognition as FaceRec
import PIL.Image  as MyPilImg
import PIL.ImageDraw as MyPilImgDraw
```

2. Load image to Numpy array

```
ImgeArray= FaceRec.load_image_file("Pics/me.png" )
```

3. Detect face location by face-detection library

```
face_locations=FaceRec.face_locations(ImgeArray)
# top , right , bottom , left => face_location
```



4. Load pillow image from Numpy array

```
pil_image : MyPilImg.fromarray(ImgeArray)
#pil_image: MyPilImg.Image =MyPilImg.open("Pics/me.png")
```

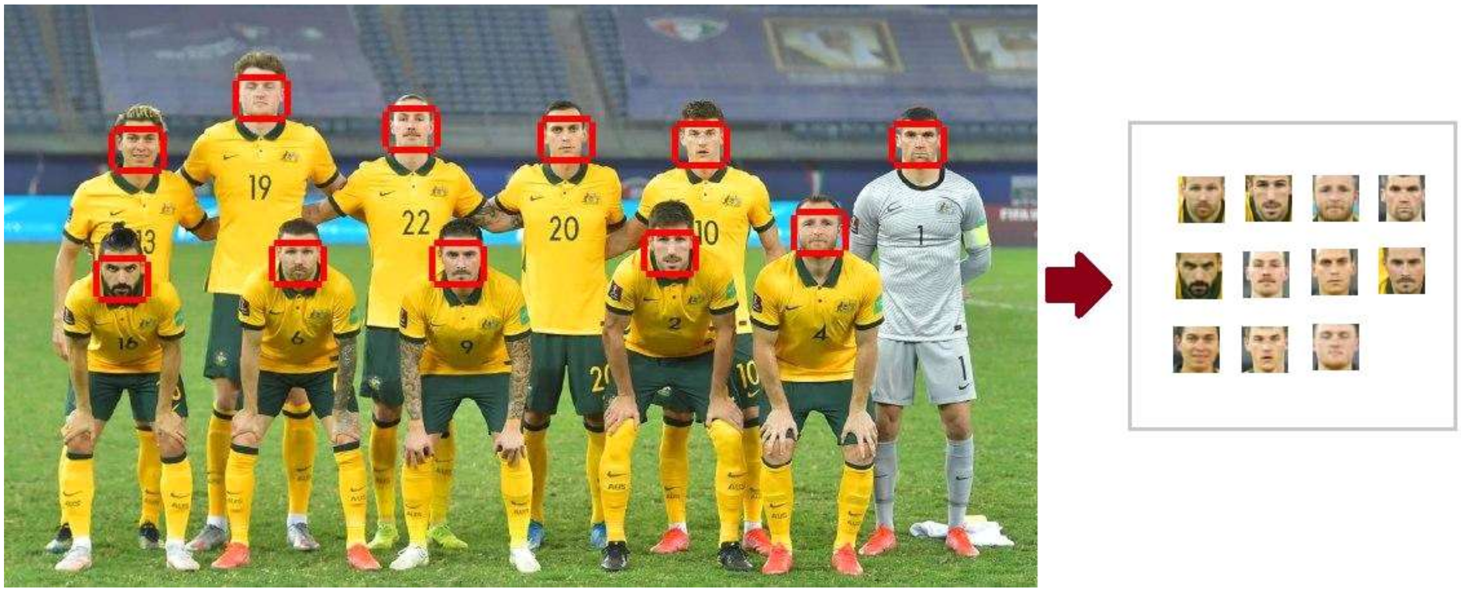
5. Find location of faces in image & draw rectangle

```
#------------------------------

for face_loc in face_locations:

    top , right , bottom , left = face_loc

    #------------------------------

    draw = MyPilImgDraw.Draw(pil_image)

    draw.rectangle( [ right , top , left , bottom ] , outline="yellow", width=5)
```

6. Show and save image

```
pil_image.show()
pil_image.thumbnail((700,700)) #Optional
pil_image.save("Out/me_face_dt.png" )
```

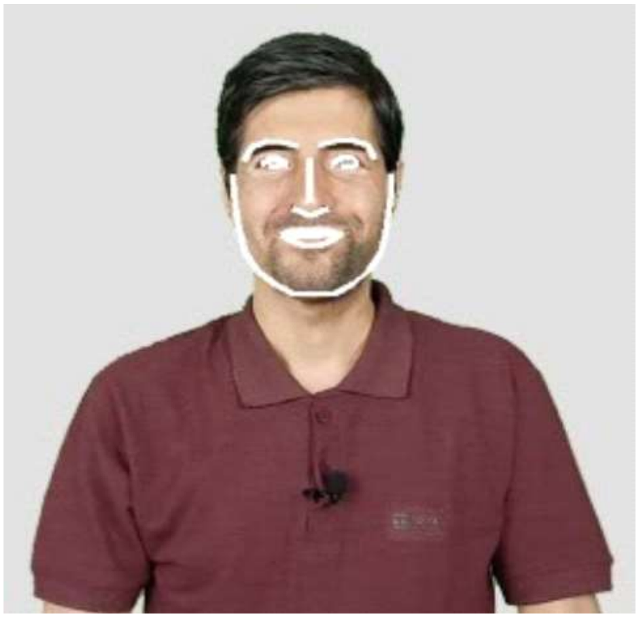
7. Save each face in dedicated image file



```
#------------------------------

import face_recognition as FaceRec
import PIL.Image as MyPilImg

#-----------load image ---------------
ImageArray=FaceRec.load_image_file("Pics/school.jpg")
#-------- detect faces -------------------
face_locations=FaceRec.face_locations(ImageArray)
#-------- Load pillow image from Numpy array --------
#~~PilImage=MyPilImg.fromarray(ImageArray)~~
#-------- Find location of faces in image & draw rectangle ----

n=1
```

```
for face in face_locations:
    # ----------------
    top,right,bottom,left=face
    # ----------------
    FaceImg= ImageArray[ top:bottom , left:right ]
    PilImage=MyPilImg.fromarray(FaceImg)
    # ----------------
    PilImage.show()
    # ------------------------

   PilImage.save("Out/face_"+str(n)+".jpg") #=> face_1.jpg , ...
   n=n+1
```

7. Detecting face landmarks (facial features)



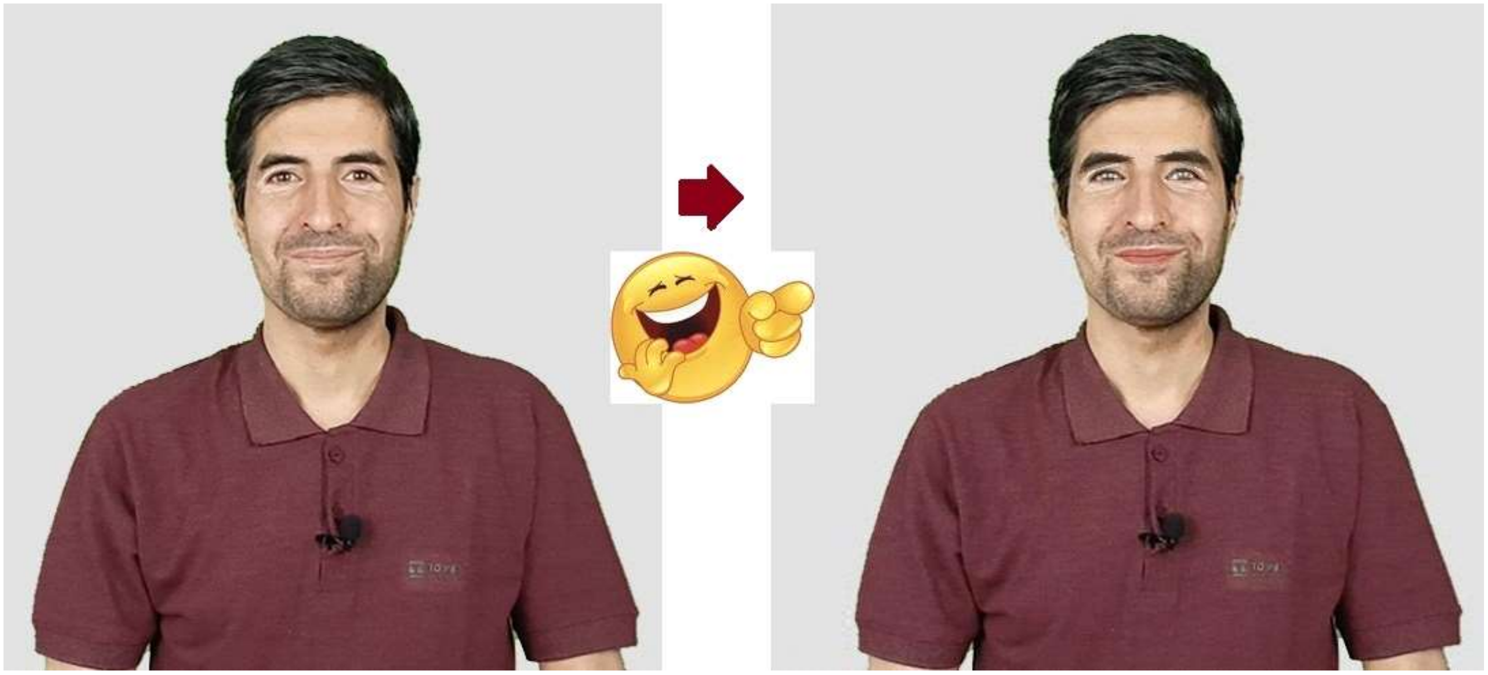
facial features are about 9 items:

- chin
- left_eyebrow, right_eyebrow
- nose_bridge, nose_tip
- left_eye, right_eye
- top_lip, bottom_lip

```
#========================================

import face_recognition as FaceRec

import PIL.Image  as MyPilImg
import PIL.ImageDraw as MyPilImgDraw

#------------load image ----------------

ImageArray=FaceRec.load_image_file("Pics/me.jpg")

# --Find all facial features in all the faces in the image

face_landmarks_list = FaceRec.face_landmarks(ImageArray)

#-------- Load pillow image from Numpy array --------

PilImage=MyPilImg.fromarray(ImageArray)

#-----------------------------

draw = MyPilImgDraw.Draw(PilImage)

#----------Find all facial features and draw lines -------------------

for face_landmarks in face_landmarks_list:

    # ---- Print the name of each facial feature in this image---

    for facial_feature in face_landmarks.keys():

         print("Feature name: ", facial_feature)

    # ----- Draw each facial feature in the image with a line---

    for facial_feature in face_landmarks.keys():

        draw.line( face_landmarks[facial_feature] , width=5,fill="white" )
```

```
#----Show and save image -------------------------

PilImage.show()
PilImage.save("Out/me_landmarks.jpg" )
```

8. Add makeup to faces!



```
import face_recognition as FaceRec

import PIL.Image as MyPilImg

import  PIL.ImageDraw as MyPilImgDraw
```

```
#-----------load image ---------------
ImageArray=FaceRec.load_image_file("Pics/me_2021.jpg")
#-------- Find all facial features in all the faces in the image----
face_landmarks_list=FaceRec.face_landmarks(ImageArray)
#-------- Load pillow image from Numpy array --------
PilImage=MyPilImg.fromarray(ImageArray)
#--------------
draw = MyPilImgDraw.Draw(PilImage, 'RGBA')
#----------------------------------
#print(face_landmarks_list)
for face_landmark in face_landmarks_list:
    #--------- add rouge makeup for lips!--------
    draw.polygon(face_landmark ["top_lip"], fill=(150, 0, 0, 64))
    draw.polygon(face_landmark["bottom_lip"], fill=(150, 0, 0, 64))
    #draw.line(face_landmark ['top_lip'], fill=(150, 0, 0, 30), width=2)
    #draw.line(face_landmark ['bottom_lip'], fill=(150, 0, 0, 30), width=2)
   # ---------Thicker eyebrows ! -------------
   draw.polygon(face_landmark ['left_eyebrow'], fill=(68, 54, 39, 128))
   draw.polygon(face_landmark ['right_eyebrow'], fill=(68, 54, 39, 128))
   #draw.line(face_landmark ['left_eyebrow'], fill=(68, 54, 39, 150), width=5)
```

```
  #draw.line(face_landmark ['right_eyebrow'], fill=(68, 54, 39, 150), width=5)
  # -------- Apply some eyeliner-----------------
  draw.line(face_landmark ['left_eye']   , fill=(0, 0, 0, 110), width=2)
  draw.line(face_landmark ['right_eye']  , fill=(0, 0, 0, 110), width=2)

   # --------sparkle on the eyes------------
  draw.polygon(face_landmark ['left_eye'], fill=(255, 255, 255, 90))
  draw.polygon(face_landmark ['right_eye'], fill=(255, 255, 255, 90))


#----Show and save image ------------------------
PilImage.show()
PilImage.save("Out/me_makeup.jpg" )
```