

Activité 4 Photographie - **Filtrage des images avec Python et PIL****I - Rappels sur les couches**

Image en nuances de gris : 1 couche par pixel (donc 1 octet)
 Images en couleurs : 3 couches RGB par pixel (donc 3 octets)
 Images en couleur avec transparence : 4 couches RGBA, A pour Alpha, la transparence.

II - Obtenir des informations sur l'image

```
01 from PIL import Image # Importe l'objet Image de gestion des images
02 ref_image = Image.open("Nom image.png") # Ouvre l'image fournie et la relie à ref_image
03 largeur = ref_image.width # stocke le nombre de pixels en largeur
04 hauteur = ref_image.height # idem mais en hauteur
05 couches = ref_image.getbands() # agit sur ref_image pour obtenir les couches
06
07 nombre_couches = len(couches) # fonction len() pour avoir le nombre de couches
08
09 print("Taille de l'image : ", largeur, " x ", hauteur) # Affichage des nombres de px
10
11 ref_image.close() # Ferme cette image interne à Python
```

III - Obtenir les valeurs RGB d'un pixel : getpixel(coordonnées)

```
01 from PIL import Image # Importe l'objet Image de gestion des images
02 ref_image = Image.open("Nom image.png") # Ouvre l'image fournie et la relie à ref_image
03 largeur = ref_image.width # stocke le nombre de pixels en largeur
04 hauteur = ref_image.height # idem mais en hauteur
05 couches = ref_image.getbands() # agit sur ref_image pour obtenir les couches
06 x = 50 # on voudra le pixel en x=50 en largeur
07 y = 100 # on voudra le pixel en y = 100 en hauteur
08
09 if len(couches) == 3: # si image RGB
10     r, g, b = ref_image.getpixel( (x,y) ) # récupère les 3 valeurs du pixel (x,y)
11
12 elif len(couches) == 4: # sinon si image RGBA
13     r, g, b, a = ref_image.getpixel( (x,y) ) # récupère les 4 valeurs du pixel (x,y)
14
15 print(r, "-", g, "-", b) # Affiche les valeurs séparées par -
16 ref_image.close() # Ferme cette image interne à Python
```

IV - Modifier les valeurs RGB d'un pixel : putpixel(coordonnées, nouvelles valeurs)

Ici, on veut supprimer le rouge d'un pixel en particulier (le (50, 100))

```
01 from PIL import Image # Importe l'objet Image de gestion des images
02 ref_image = Image.open("Nom image.png") # Ouvre l'image fournie et la relie à ref_image
03 largeur = ref_image.width # stocke le nombre de pixels en largeur
04 hauteur = ref_image.height # idem mais en hauteur
05 couches = ref_image.getbands() # agit sur ref_image pour obtenir les couches
06 x = 50 # on voudra le pixel en x=50 en largeur
07 y = 100 # on voudra le pixel en y = 100 en hauteur
08
09 if len(couches) == 3: # si image RGB
10     r, g, b = ref_image.getpixel( (x,y) ) # récupère les 3 valeurs du pixel (x,y)
11     ref_image.putpixel( (x,y), (r,g,b) ) # supprime le rouge de ce pixel
12
13 elif len(couches) == 4: # sinon si image RGBA
14     r, g, b, a = getpixel( (x,y) ) # récupère les 4 valeurs du pixel (x,y)
15     ref_image.putpixel( (x,y), (r,g,b,a) ) # supprime le rouge de ce pixel
16
17 print(r, "-", g, "-", b) # Affiche les valeurs RGB séparées par des tirets
18 ref_image.close() # Ferme cette image interne à Python
```

V - Lire et modifier tous les pixels d'une image

```
01 from PIL import Image                # Importe l'objet Image de gestion des images
02
03 def filtrage(r, g, b):
04     return (r, 0, 0)                 # Exemple de filtre rouge
05
06 ref_image = Image.open("Nom image.png") # Ouvre l'image fournie et la relie à ref_image
07 largeur = ref_image.width            # stocke le nombre de pixels en largeur
08 hauteur = ref_image.height           # idem mais en hauteur
09 couches = ref_image.getbands()       # agit sur ref_image pour obtenir les couches
10
11 for y in range(hauteur):              # pour chaque ligne y de l'image (du 0 au maximum)
12     for x in range(largeur):          # pour chaque colonne x de l'image (du 0 au maximum)
13
14         if len(couches) == 3:         # si image RGB
15             r, g, b = ref_image.getpixel( (x,y) ) # récupère les 3 valeurs en (x,y)
16             r, g, b = filtre_a(r,g,b) # Modifie RGB
17             ref_image.putpixel( (x,y), (r,g,b) ) # Place nouveau RGB sur ce pixel
18
19         if len(couches) == 4:         # si image RGB
20             r, g, b, a = ref_image.getpixel( (x,y) ) # récupère les 3 valeurs en (x,y)
21             r, g, b = filtrer(r, g, b) # Modifie RGB
22             ref_image.putpixel( (x,y), (r,g,b,a) ) # Place nouveau RGBA sur ce pixel
23
24 ref_image.show()                      # Affiche l'image
25 ref_image.save("Nom image.png")      # Sauvegarde sur le disque sous le nom fourni
26 ref_image.close()                    # Ferme cette image interne à Python
```

Remarque : pourquoi utiliser une fonction d'une seule ligne ? Simplement car de cette façon, elle est réutilisable à la fois pour le cas 3 couches et le cas 4 couches.

BILAN DES FONCTIONS ET METHODES RENCONTREES

Image.open(...) _____

reference.getbands() _____

reference.getpixel((x,y)) _____

reference.putpixel((x,y), couleurs) _____

reference.save(...) _____

reference.show() _____

reference.close() _____