



LA PHOTOGRAPHIE NUMÉRIQUE PYTHON

Contenus et capacités

Contenus

Traitement d'image.

Capacités attendues

Traiter par programme une image pour la transformer en agissant sur les trois composantes de ses pixels.

Activité

Sensibiliser les élèves à la démarche algorithmique permettant de « parcourir » les pixels d'une image.

Durée indicative

1 heure.

Note d'intention

Cette activité se situe dans la progression après avoir expliqué aux élèves les notions de photosites, pixels, composantes de pixels, etc., et avant de modifier une image existante (transformation en niveau de gris, filtre négatif, etc.).

En partant du code permettant de colorer un pixel, les élèves sont amenés progressivement à la notion de boucle afin de colorer une ligne entière, puis de 2 boucles imbriquées afin de colorer l'image entière. L'activité se termine par l'usage de fonctions permettant d'obtenir une image de la couleur de son choix, voire avec les dimensions de son choix. On peut sensibiliser les élèves à l'utilisation de la console pour obtenir des résultats différents sans avoir à exécuter à nouveau le code, d'où l'intérêt d'une fonction avec paramètres.

Les élèves les plus rapides peuvent être occupés avec la recherche d'un dégradé linéaire, dont une correction possible - sans fonction pour ne pas compliquer la tâche - est donnée avec ce code générant un dégradé vertical du rouge au bleu :

```
img = Image.new("RGB", (200, 100))
for abscisse in range(200):
    for ordonnee in range(100):
        img.putpixel((abscisse, ordonnee), (((100-ordonnee)*255)//100, 0, (ordonnee*255)//100))
img.show()
```

Voici comment il est possible de représenter par exemple un écran de 64 pixels.

	0	1	2	3	4	5	6	7
0								
1								
2								
3								
4								
5								
6								
7								

À l'aide de l'éditeur Python, créer un nouveau fichier et recopier les 2 lignes ci-dessous.

```
from PIL import Image
img = Image.new("RGB", (8,8))
```

La première permet de charger la librairie et la seconde de créer une image de 64 pixels (pour l'instant tous de couleur noire).

Ajouter cette ligne qui permet de colorer le pixel en haut à gauche (de coordonnées (0,0)) en rouge (on précise en effet que le sous-pixel rouge est à 255, le vert à 0 et le bleu à 0).

```
img.putpixel((0,0), (255,0,0))
```

Et enfin celle-ci pour visualiser l'image : (ne pas hésiter à zoomer pour bien voir le pixel rouge).

```
img.show()
```

Réaliser les étapes suivantes, à faire valider par l'enseignant une par une :

1. Modifier le programme pour colorer en rouge uniquement le deuxième pixel de la première ligne puis uniquement le troisième.
2. Modifier le programme pour colorer en rouge toute la première ligne, en trouvant la méthode la plus concise.
3. Modifier à nouveau le programme pour colorer en rouge uniquement la deuxième ligne, puis uniquement la troisième ligne.

4. Trouver comment colorer en rouge toutes les lignes de l'image, de la façon la plus simple possible.
5. Afin de pouvoir facilement changer de couleur, créer une fonction `creation_image` utilisant un paramètre nommé `couleur` qui contiendra le code actuel du programme (uniquement la première ligne figure sur le code ci-dessous).

```
def creation_image (couleur):  
    img = Image.new("RGB", (8, 8))
```

Une fois votre code exécuté, l'appel en console de `creation_image ((255,0,0))` devra afficher une image rouge mais `creation_image ((0,255,0))` devra afficher une image verte, ...

6. Ajouter deux paramètres à la fonction, longueur et largeur, pour pouvoir changer le nombre de pixels à la demande.
7. Pour aller plus loin : réaliser une image avec un dégradé.

Retrouvez éduscol sur :

