1 Images numériques : format vectoriel ou bitmap

Exercice 1 *Généralités*

1. Ouvrir les images `Plan-Lyon-Metro-Tramway.png` et `Plan-Lyon-Metro-Tramway.svg`[a] avec la visionneuse d'images par défaut ou avec le navigateur Firefox. Voyez-vous une différence ? Dans Firefox, faire un clic droit sur chaque image. Les options disponibles sont-elles les mêmes ? Cliquez sur *code source* pour l'image au format `svg` et sur *informations sur l'image* pour l'image au format `png`.

Faire une recherche documentaire sur les termes suivants : *format de fichier informatique, fichier texte, fichier binaire.* Classez les formats `svg` et `png` dans leur catégorie.

2. Pour les deux images, faites un zoom sur le mot Part de Part-Dieu. Normalement vous devriez obtenir un effet de pixellisation pour l'image au format `png` mais pas pour l'image au format `svg` :

a. Image de David Arthur

3. Lorsqu'on manipule des images numérisées dans des fichiers informatiques on parle souvent de *pixel*, de *définition* de 800 × 600 pixels d'une image, de *résolution* de 72 pixels per inch . . .

Définir ces termes. On pourra visiter les pages web suivantes :

- https://interstices.info/jcms/c_41905/nom-de-code-binaire
- https://interstices.info/jcms/c_43823/tout-a-un-reflet-numerique

4. Expliquez le phénomène de *pixellisation*. Pour quelles images est-il plus pertinent d'utiliser un format d'image vectoriel ? bitmap ? Citez au moins trois formats d'images vectoriels et trois formats d'images bitmap.

Exercice 2 *SVG un format vectoriel*

Le code ci-dessous permet de représenter un disque noir en SVG (Scalable Vector Graphics) qui est un format ouvert d'image vectorielle spécifié par le World Wide Web Consortium (http://www.w3.org/). Le W3C est un organisme de normalisation à but non-lucratif fondé par Tim Berners-Lee créateur du HTML, du protocole HTTP, des adresses Web. Le W3C est chargé de promouvoir les technologies du Web telles que HTML, XML, CSS, PNG, SVG . . .

1. Ouvrir le fichier `disquenoir.svg` avec le navigateur Firefox. Un autre exemple d'image au format `svg` est donné en page 5 du fichier `document1.pdf`.

2. Quelle est la ligne de code permettant de tracer le disque ? Modifier cette ligne pour tracer un carré rouge de côté 300 pixels. On pourra consulter https://www.w3schools.com/html/html5_svg.asp.

3. Les lignes 1 à 4 représentent l'en-tête de l'image avec la spécification du format. Que représentent les lignes 6 et 8 ?

4. Agrandir ou réduire l'image avec le zoom. Le contour du disque est-il moins régulier ?

```
1  <?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
2  <svg width="800px"
3       height="800px"
4       xmlns="http://www.w3.org/2000/svg">
5  <circle cx="400px" cy="400px" r="100px" fill="black" />
6  <title>Disque noir en SVG </title>
7   <desc>
8  <Creator>Lycee du Parc </Creator>
9  </desc>
10 </svg>
```

disquenoir.svg

Exercice 3 *PBM un format bitmap*

On donne ci-dessous le code de la représentation bitmap d'une image représentant un disque noir sur un fonds blanc. C'est une image matricielle de format PBM qui permet de coder dans un fichier texte des images en noir et blanc.
Les lignes précédées d'un # sont des commentaires.

1. Ouvrir le fichier `disquebnoir.pbm` avec un éditeur de texte comme Notepad+ ou Scite. Que représentent les valeurs de la ligne 4 ?

2. Ouvrir le fichier `disquebnoir.pbm` avec le logiciel `Gimp`. Sachant que la résolution moyenne d'un écran 4 : 3 de diagonale 19 pouces (1 pouce mesure $0,0254m$), avec une définition de 1024×768 est d'environ 72 ppi, expliquer la taille de l'image affichée.

   Agrandir cette image avec +. L'effet observé sur le contour du disque est une *pixellisation*.

3. Remplacer P1 par P2 sur la ligne 1 puis rafraîchir l'image avec `Fichier>Rétablir`. Que se passe-t-il ?Remettre P1 pour la suite.

4. Modifier l'image pour représenter un carré noir de côté 9 pixels puis la lettre I dans un cadre de dimensions $Largeur \times Hauteur = 5 \times 10$. Peut-on représenter un carré rouge avec ce format ?

   On pourra consulter la spécification de ce format de fichier bitmap sur la page web http://fr.wikipedia.org/wiki/Portable_pixmap#Fichier_ASCII.

```
1  P1
2  #Createur : Lycee du Parc
3  #Titre : Disque noir
4  13 13
5  0000000000000
6  0000000000000
7  0000111110000
8  0001111111000
9  0011111111100
10 0011111111100
11 0011111111100
12 0011111111100
13 0011111111100
14 0001111111000
```

```
0000111110000
0000000000000
0000000000000
```

disquenoir.pbm

2 Codage des couleurs dans un format bitmap

Dorénavant nous travaillerons principalement sur des formats `bitmap`.

Exercice 4 *Représentation en niveaux de gris*

Le fichier `niveauxgrisP2.pgm` contient la représentation bitmap d'une image en niveaux de gris au format P2. C'est un fichier texte. Les pixels sont codés ligne par ligne, de haut en bas et de gauche à droite.

Un autre exemple d'image au format `pbm` est donné en page 4 du fichier `document1.pdf`.

Bloc-Note

Pour représenter une image en *niveaux de gris*, on choisit une valeur maximale, par exemple 255, pour exprimer les niveaux de gris et on associe à chaque pixel un nombre compris entre 0 et 255 pour coder les différentes nuances de gris.

1. Ouvrir le fichier `niveauxgrisP2.pgm` avec un éditeur de texte pour accéder au code puis avec `Gimp` pour afficher l'image. Dans le menu Image, sélectionner Propriétés de l'image et noter l'espace de couleurs.
2. Lire l'en-tête qui occupe les lignes 1 à 4 avec le nombre magique P2 à la ligne 1, les dimensions de l'image (largeur puis hauteur séparées par un espace) à la ligne 3 et le nombre maximal de niveaux de gris à la ligne 4.
3. Les pixels de l'image sont codés de la ligne 5 à la ligne 13.

   Comment est codé le noir ? le blanc ? Et un gris foncé ? un gris clair ?

Exercice 5

1. Ouvrir avec un lecteur de `pdf` le fichier `document1.pdf`, lire les pages 7 à 10 et répondre aux questions suivantes [1] :
   - Quel système de représentation des couleurs est utilisé fréquemment en informatique ?

1. On pourra compléter avec les informations trouvées sur les pages web citées dans l'exercice 1

- Combien de couleurs sont-elles disponibles avec cette représentation ?
- Calculer le poids en octets d'une image couleur de définition 600 × 500 pixels dont la profondeur est de 24 bits ?
- Quelle est la différence entre la *synthèse additive* et la *synthèse soustractive* des couleurs ? Pourquoi les écrans utilisent-ils un système de représentation RVB alors que les imprimantes jets d'encre ont besoin de quatre couleurs CMYB ?

2. Le fichier `rgbP3.ppm` contient une palette de $L \times H = 3 \times 2$ couleurs représentées en bitmap RVB dans un fichier texte.

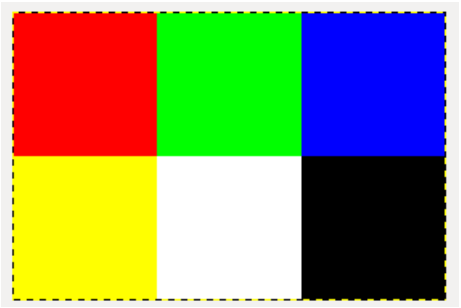
L'en tête du fichier occupe les lignes 1 à 4 : les caractères magiques P3 en ligne 1, un commentaire en ligne 2, la largeur puis la hauteur en ligne 3 (séparées par un espace), la valeur maximale d'un canal en ligne 4.

a. Ouvrir le fichier avec un éditeur de texte. Repérer l'en tête et le codage des neuf pixels. Ouvrir ensuite le fichier avec Gimp, agrandir avec + et dans le menu `Image`, sélectionner `Propriétés de l'image` puis noter l'espace de couleurs.

b. Tester les représentations RVB de différentes couleurs avec l'outil disponible sur `https://www.w3schools.com/colors/colors_picker.asp`.

c. Sachant que le jaune s'obtient en mélangeant à parts égales du rouge et du vert, comment peut-on obtenir de l'orange ?

Dans un éditeur de texte, créer un fichier représentant selon le format PPM P3 un carré orange de dimensions 10×10 pixels. Ce type de codage vous semble-t-il efficace ? Imaginer une méthode de compression du codage.



Exercice 6

1. Ouvrez l'image `lena.png` avec `Gimp`, sélectionner l'outil `Propriétés de l'image` du menu `Image` et noter sa définition en pixels, sa résolution en ppp, son poids en octets et son espace de couleurs. Que représentent ces différentes caractéristiques ?

2. Sélectionner l'outil `Fenêtres>Fenêtre ancrable>Canaux` et choisissez un affichage avec un seul canal couleur(Rouge, Vert ou Bleu) comme ci-dessous :

Tester différentes combinaisons de 2 canaux. Que se passe-t-il si on désactive les trois canaux ? Quel lien existe-t-il entre le nombre de canaux et la *profondeur* d'une image définie dans l'exercice précédent ?

3. Dans le menu `Fichier` sélectionner l'outil `Exporter vers` puis enregistrer une copie compressée au format JPEG du fichier `lena.png` sous le nom `lena-compress.jpg`. Choisir un taux de 10. Comparer les tailles en octets et la qualité du fichier obtenu et du fichier source.

Faire une recherche documentaire sur la compression numérique et le format JPEG. Lire le paragraphe *Trucs pour la mémoire* de la page https://interstices.info/jcms/c_41905/nom-de-code-binaire et répondre aux questions suivantes :

- Pourquoi compresse-t-on des images ou d'autres objets numérisés ?
- Pourquoi peut-on se contenter de compression avec perte pour des images ou des sons numérisés ?
- La compression est-elle surtout utilisée pour les images bitmap ou les images vectorielles ?
- Ouvrir un *email* contenant une image en pièce jointe et faire afficher la source. Le message est un fichier texte dans lequel l'image est codée dans une suite de caractères incompréhensible. Comment s'appelle ce codage ? Quelles sont ses utilisations ? S'agit-il d'une méthode de compression des données ?

3 Premiers traitements d'image en Python

Bloc-Note

`PIL` (Python Imaging Library) est un module (ou bibliothèque) Python permettant de traiter des images. `PIL` ne fait pas partie de la bibliothèque standard de Python : il s'agit d'une bibliothèque supplémentaire (libre) qu'il faut installer. Pour se faire il faut avoir une connexion internet et dans l'interpréteur Python de l'IDE `Pyzo` il suffit de saisir une des deux lignes suivantes :

```
>>> pip install pillow
>>> conda install pillow
```

l'installation se fera automatiquement. La documentation est disponible à partir de la page web https://pillow.readthedocs.io/en/4.1.x/.

Nous allons utiliser le module `PIL`, ou plutôt un sous-ensemble de ce module appelé `Image`, pour réaliser différentes opérations sur des images.

Principales fonctionnalités du module `Image` de `PIL`

`nom_image = Image.open("chemin_fichier")`	ouverture d'un fichier image
`nom_image.putpixel((x,y),(r,g,b))`	écriture de la valeur (r,g,b) dans le pixel (x,y)
`nom_pixel = nom_image.getpixel((x,y))`	lecture du pixel de coordonnées (x,y)
`r,g,b = nom_image.split()`	récupère les composantes (r,g,b) de l'image
`pixel = nom_image.load()`	charge le tableau de pixels dans une variable accès en lecture et écriture avec pixel[x, y]
`nom_image.size`	taille de l'image sous la forme (Largeur,Hauteur)
`nom_image.format`	format de l'image (jpeg, png ...)
`nom_image.mode`	mode de l'image : 'L' en niveaux de gris, 'RGB' en couleurs
`nom_image.save("chemin_fichier")`	sauvegarde d'une image dans un fichier
`nom_image = Image.new('RGB',(L,H))`	création d'une image 'RGB' de dimensions (Larg.,Haut.)

Exercice 7 *découverte de PIL*

Nous allons travailler sur des images placées sous la licence `Creative Commons CC0 1.0` par le Metropolitan Museum of Art de New York.

Dans le dossier `ressources` se trouvent le fichier `holyfamily.jpg` représentant le tableau « *The Holy Family with Saint Elizabeth, Saint John, and a Dove* » de *Peter Paul Rubens* et le fichier `cypres.jpg` représentant le tableau « *Cypres* » de *Vincent van Gogh*.

Ces fichiers ont été téléchargés à partir du moteur de recherche d'images sous licence Creative Commons : https://ccsearch.creativecommons.org/.

1. Quelles sont les utilisations permises par la licence `Creative Commons CC0 1.0` ?

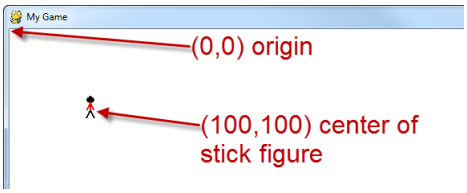
2. Avec l'éditeur de `Pyzo`, créer un script `Images-Activite1.py`, récupérer les lignes suivantes dans le fichier `cadeau.py` puis exécuter avec l'option `Run as script`. Les variables `im1` et `im2` vont pointer sur les représentations des deux fichiers images chargés en mémoire vive. `Python` ne manipulera que ces objets, les fichiers sur le disque dur ne seront modifiés que s'ils sont écrasés par une instruction spécifique.

```
from PIL import Image           # import du module PIL

im1 = Image.open("holyfamily.jpg") #stockage d'un fichier image en memoire vive
im2 = Image.open("cypres.jpg")   #stockage d'un fichier image en memoire vive
(L1, H1) = im1.size              #dimensions de l'image (Largeur, Hauteur)
(L2, H2) = im2.size
print("Image 1 : Mode :", im1.mode ,"Largeur = ", L1, "Hauteur = ", H1)
print("Image 1 : Mode :", im2.mode, "Largeur = ", L2, "Hauteur = ", H2)
```

Bloc-Note

Dans une image, les pixels sont repérés depuis une origine $(0,0)$ située dans le coin supérieur gauche avec un axe des abscisses horizontal orienté positivement de gauche à droite et un axe des ordonnées vertical orienté positivement de haut en bas.



3. Les deux images sont presque de dimensions 486×624 donc nous pouvons les découper pour leur donner cette dimension sans trop les altérer. Récupérer la fonction suivante dans `cadeau.py`.

```
def decoupage(imsource, L, H):
    (l, h) = imsource.size
    imbut = Image.new(imsource.mode, (L, H)) #creation d'une nouvelle image en
         memoire vive
    if L <= l and H <= h:            #test de possibilite du decoupage
        pixelsource = imsource.load() #tableau d'acces aux pixels de la source
        pixelbut = imbut.load()      #tableau d'acces aux pixels du but
        for y in range(H):           #pour chaque ligne
            for x in range(L):       #pour chaque colonne
                pixelbut[x, y] = pixelsource[x, y]  #copie de pixel
        return imbut                 #retourne l'image but
    return None                      #ne retourne rien Sinon
```

Appliquer cette fonction aux deux images puis enregistrer les images découpées avec la méthode `save`.

Comparer la taille des fichiers obtenus après découpage avec celles des fichiers sources. Comment expliquer ce paradoxe ?

```
In [23]: im3 = decoupage(im2, 486, 624)

In [24]: im3.save("cypres-486x624.bmp")

In [25]: im4 = decoupage(im1, 486, 624)

In [24]: im4.save("holyfamily-486x624.bmp")
```

Exercice 8 *Filtrage par canal de couleur*

1. Compléter le code de la fonction, `filtre_composante(imsource, canal)`.

```
def filtre_composante(imsource, canal):
    """Retourne une image but obtenue en mettant a 0 les valeurs des pixels de
         imsource selon les deux canaux (R,G,B) differents du parametre canal
         qui vaut : 0 pour 'R', 1 pour 'G' et 2 pour 'B' """
    pixelsource = imsource.load()
    (L, H) = imsource.size
    imbut = Image.new('RGB', (L, H))
    pixelbut = imbut.load()
    for y in range(H):
        for x in range(L):
            val = [0, 0, 0]
            #a completer
    return imbut
```

2. Utiliser cette fonction pour enregistrer sur le disque les fichiers `cypres-rouge.jpg"`, `cypres-vert.jpg"` et `cypres-ble` à partir de l'image contenue dans `"cypres.jpg"`.

Exercice 9 *De la couleur au niveau de gris*

1. Compléter le code de la fonction `niveau_gris(imsource)` ci-dessous. Attention la valeur d'un niveau de gris doit être un entier entre 0 et 255 :

```
def niveau_gris(imsource):
    """Retourne une image en niveau de gris obtenue a partir de l'image (R,G,B)
          imsource
    en prenant comme niveau de gris la moyenne des composantes (R,G,B)"""
    pixelsource = imsource.load()
    (L, H) = imsource.size
    imbut = Image.new('L', (L, H)) #imbut en niveaux de gris
    pixelbut = imbut.load()
    for y in range(H):
        for x in range(L):
            #a completer
    return imbut
```

2. Utiliser cette fonction pour enregistrer sur le disque le fichier `"cypres-gris.jpg"` à partir de l'image contenue dans `"cypres.jpg"`.

4 Enjeux sociétaux et droits du numérique

Exercice 10 *Questionnements*

1. Répondre aux questions de ce QCM sur le droit à l'image : https://dane.ac-lyon.fr/spip/QCM-Droits-a-l-image
2. L'image `Plan-Lyon-Metro-Tramway.svg` a été téléchargée sur la page https://commons.wikimedia.org/wiki/File:Plan-Lyon-Metro-Tramway.svg.

   David Arthur, l'auteur de cette image l'a placé sous la licence Creative Commons Attribution - Partage dans les Mêmes Conditions 2.0 Canada. Comment peut-on utiliser cette image ?
3. Peut-on insérer une photo qu'on a prise de la Joconde dans son site internet ? et une photo d'un camarade ?
4. Un enseignant peut-il publier des photos des travaux de ses élèves sur son blog ou sur le site de l'établissement ?
5. Un enseignant peut utiliser des images téléchargées dans un support pédagogique diffusé aux élèves ou sur le site de l'établissement ?

Des liens utiles :

- http://eduscol.education.fr/internet-responsable/ressources/legamedia/telechargement-et-diffus html
- http://creativecommons.fr/licences/les-6-licences/

Exercice 11 *Sujets d'exposés*

Par groupe de trois, traiter l'un des sujets suivants sous forme de présentation numérique (diaporama Impress, page web HTML, prezi ...) que vous devrez présenter en dix minutes devant la classe au cours du premier trimestre.

Pour la réalisation de diaporama avec Impress, la ressource suivante est très bien : https://dane.ac-lyon.fr/spip/Video-Prise-en-main-d-Impress.

- Droit à l'image dans le numérique : cadre légal et exceptions.

  Ressources : http://eduscol.education.fr/internet-responsable/ressources/legamedia/image-et-videc html
- Puis-je insérer n'importe quelle image téléchargée dans un article de mon blog ?

  Ressources : http://eduscol.education.fr/cdi/res/ressource-enseignement/banques_dimages_lib
- Droit d'auteur : définition des droits patrimoniaux et moraux.

  Ressources : https://dane.ac-besancon.fr/les-droits-dauteur/
- Pourquoi publier une production numérique sous une licence Creative Commons ?

  Ressources : http://creativecommons.fr/ et http://eduscol.education.fr/internet-responsable/ressour legamedia/contenus-ouverts-ou-libres.html
- Les formats PNG (bitmap) et SVG (vectoriel) sont des *formats ouverts* de fichier image. Qu'est-ce-qu'un *format ouvert* ?

  Ressources : http://www.april.org/formats-ouverts-pour-quoi-faire.
- Comment fonctionne l'Hadopi ?

  Ressources : http://eduscol.education.fr/internet-responsable/ressources/legamedia/telechargement html.