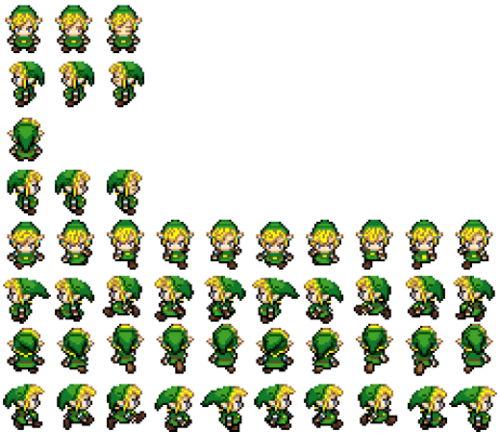
TP3 : La gestion des sprites

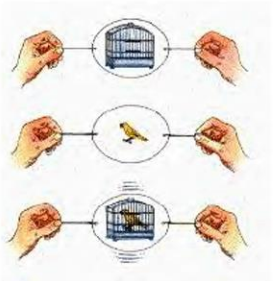
Table des matières :

1. Les sprites

Dans le domaine des jeux vidéos, un ***sprite*** (***lutin*** en anglais) est un élément graphique animé (personnages, objets, …). Le ***sprite sheet*** représente l'ensemble des mouvements du sprite sous la forme d'une matrice d'images.

Exemple d'un sprite sheet :



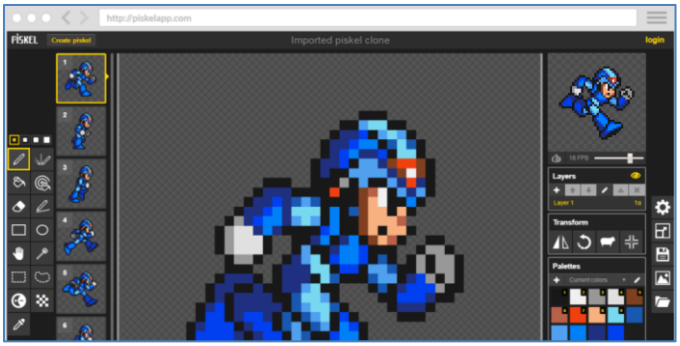


Pour réaliser l'animation d'un sprite, il faut faire défiler les images à l'écran avec une vitesse supérieure à la persistance rétinienne. La persistance rétinienne est le phénomène qui nous permet d'observer un défilement d'images fixes sans voir les coupures entre celles-ci (l'œil permet de voir à peu près une image toutes les 1/25ème de secondes).

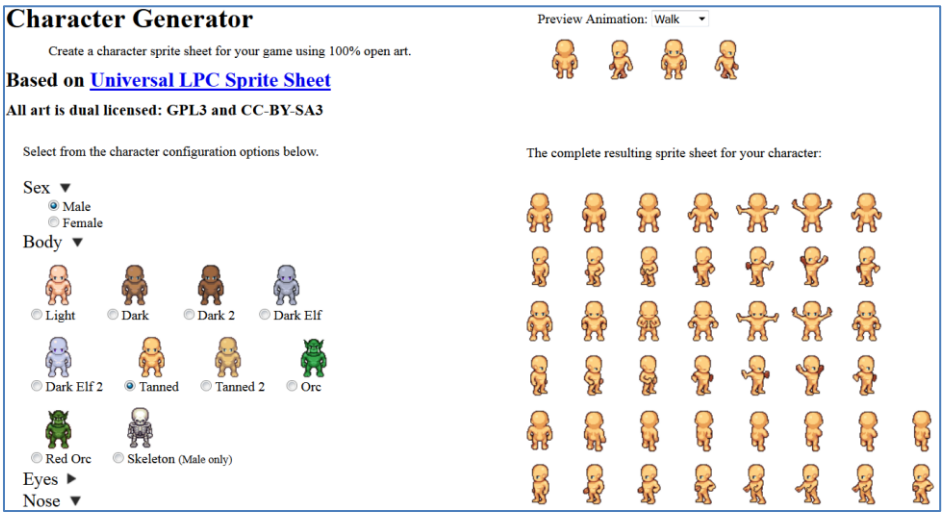
L'animation consiste donc à gérer le défilement des images.

En matière de sprites, de nombreuses ressources sont disponibles sur le web. Mais attention aux droits d'auteur, vous devez bien identifier les licences avant de pouvoir utiliser les sprites mis à disposition.

Pour créer vos propres sprites, il existe des logiciels générateurs de sprites en ligne :
https://www.piskelapp.com/



Ou encore des solutions de personnalisation de sprite sous licence GPL3 et CC-BY-SA3 :
http://gaurav.munjal.us/Universal-LPC-Spritesheet-Character-Generator/#

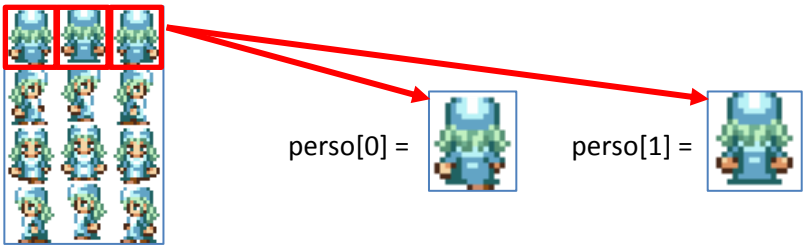


2. Animation d'un sprite

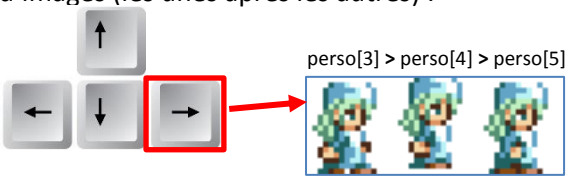
Pour réaliser notre première animation avec Processing, nous allons travailler avec le sprite sheet fourni dans le dossier ***3-ressources*** (« ***sprite_32-36.png*** »):

Ce sprite sheet est une matrice d'images (3x4=12 images).
La taille est de 96px * 144px.
Une image fait 32px de large par 36 px de haut.

Notre programme Processing doit extraire chaque image du sprite sheet avec la méthode ***get()*** (liée à la classe ***PImage***) , puis les placer dans un tableau d'images (***PImage[] perso***).



Ensuite, en fonction de la touche du clavier sélectionnée, l'algorithme doit venir afficher la bonne série d'images (les unes après les autres) :



Exemple :

```
PImage spriteSheet;
PImage[] perso= new PImage[3];
int cpt=0;
int index=0;
int coordX_perso=0;
int coordY_perso=0;

void setup() {
  size(320, 320);
  smooth();
  spriteSheet = loadImage("sprite_32-36.png");
  for (int x = 0; x < 3; x++)
    perso[x] = spriteSheet.get(x*32, 0, 32, 36);
  imageMode(CENTER);
  coordX_perso= width/2;
  coordY_perso= height/2;
}

void draw() {
  background(255);
  if (keyPressed == false) image(perso[index], coordX_perso, coordY_perso);
  else
  {
    image(perso[index+cpt], width/2, height/2);
    if (frameCount>10)
    {
      cpt++;
      if (cpt>2) cpt=0;
      frameCount=0;
    }
  }
}
```

Travail n°1 :

Recopiez le code ci-dessus dans un sketch Processing. Testez et analysez le code à l'aide de la documentation Processing (https://processing.org/reference/)

Réalisez ensuite un programme permettant d'animer le personnage en fonction des touches directionnelles de votre clavier. Pour cela, vous devez :

- Modifier la déclaration du tableau d'images ***perso[]*** (modifier la taille)
- Modifier le ***setup()*** afin de charger toutes les images du sprite dans ***perso[]***. Pour cela, modifier la structure du code de la boucle for (utiliser deux boucles ***for*** imbriquées).
- Ajouter une méthode ***moussePressed()*** et y utiliser une structure ***switch*** pour modifier la variable ***index*** en fonction du type de flèches sélectionnées (UP=>index=0, RIGHT=>index=3, …) :
  - https://processing.org/reference/keyCode.html
  - https://processing.org/reference/switch.html

3. Gestion des déplacements

Avant de gérer les déplacements de notre personnage tout en gérant son animation, nous allons devoir restructurer le code car sinon celui-ci risque de devenir complexe. Nous devons donc introduire le concept de **programmation orientée objet (POO)**.

⇨ **Lire** les pages 96-100 et les pages 114-117 du ***FlossManual Processing*** fourni en ressource.

Voici le code de notre **classe *personnage* :**

```
class personnage {
  PImage perso[];
  PImage spriteSheet;
  int cpt_draw = 0, index = 0, cpt_img=0;

  personnage(String spriteSheetFile) {
    spriteSheet=loadImage(spriteSheetFile);
    perso = new PImage[12];
    for (int y = 0; y < 4; y++)
      for (int x = 0; x < 3; x++)
        perso[y*3+x] = spriteSheet.get(x*32, y*36, 32, 36);
  }

  void sens_marche(int sens) {
    switch(sens) {
    case 0:
      index=0;
      break;
    case 1:
      index=3;
      break;
    case 2:
      index=6;
      break;
    case 3:
      index=9;
      break;
    }
  }

  void deplace(int coordX, int coordY) {
    if (cpt_draw++ > 5) { //nbr de draw() pour incrementer cpt_img
      cpt_draw = 0;
      cpt_img++;
      if (cpt_img > 2)
        cpt_img = 0;
    }
    image(perso[index+cpt_img], coordX, coordY);
  }

  void arret(int coordX, int coordY) {
    image(perso[index+cpt_img], coordX, coordY);
  }
}
```

Travail n°2 :
Analysez le code de la classe ***personnage***.
Puis à l'aide de la classe ***personnage***, réalisez un programme permettant de déplacer et d'animer le sprite du travail n°1 en fonction des touches directionnelles de votre clavier.