|  |  |
| --- | --- |
|  | Premiers pas en Réalité augmentée |

Thème d’étude : Conception d’une vidéo en réalité augmentée

Nature de la production finale attendue : Conception d’une vidéo en réalité augmentée générée par le logiciel Processing. Cette vidéo doit présenter une visite partielle d’une salle de cours avec incrustation à partir de marqueurs d’images générées par du code.

Cahier des charges : Cette vidéo comporte 5 marqueurs.

*On donne :*

* *Le programme partiel avec détection à partir de 3 marqueurs.*
* *Les 3 marqueurs déjà paramétrés*

*On demande :*

* *Adapter le programme existant pour ajouter la gestion des marqueurs 4 et 5 pour faire apparaitre un cube, puis une sphère.*

*On exige :*

* *Un programme fonctionnel en 2 étapes* 
  + *Marqueur 4 tout d’abord avec validation par l’enseignant*
  + *Marqueur 4 et 5 avec validation par l’enseignant*

|  |  |  |  |
| --- | --- | --- | --- |
| **Conditions** | Travail dirigé  Durée : 3 h | **Moyens** | * Poste informatique sous Windows * Logiciels PROCESSING |
| **Prérequis** | Représentation de l'information | **Niveau** | Classe de Terminale S spécialité ISN |
| **Compétences** | * C1.1 Justifier dans une situation donnée, un codage numérique ou l'usage d'un format approprié, qu'un programme réalise l'action attendue... * C2.3 Développer une interface logicielle ou une interface homme-machine, un algorithme, un programme, un document ou fichier numérique... | | |
| **Eléments du programme** | * Représentation de l’information   + Structuration et organisation de l’information * Algorithmes   + Algorithmes simples | | |

**Table des matières :**

[1. Les fonctions principales 2](#_Toc360451096)

[2. Le programme commenté 2](#_Toc360451097)

[3. Modification du programme – travail demandé 6](#_Toc360451098)

[Correction : 7](#_Toc360451099)

[4. Les marqueurs utilisés 8](#_Toc360451103)

# Les fonctions principales

Le programme utilise un certain nombre de fonctions dont certaines reviennent assez souvent :

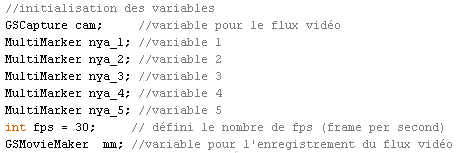
* import : permet l'importation de librairie
* void : indique que la fonction ne renvoie aucune valeur
* printIn : affiche dans la zone de texte l'évolution d'une, ou plusieurs variables
* text : fonction d'affichage de texte
* loadImage : fonction de chargement d'une image (appelle une image)
* fill (R,G,B) : permet le changement de couleur d'une fonction telle que « text »

# Le programme commenté

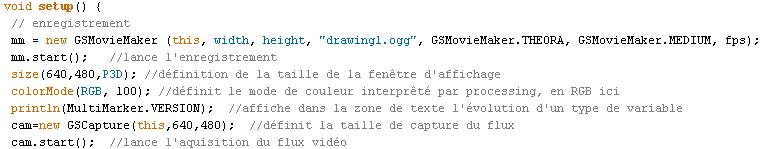
Importation des librairies (*java, opengl, nyartoolkit, gsvideo*), chaque librairie permet d'ajouter des fonctionnalités, le programme peut utiliser plus ou moins de fonction selon les librairies importées.



Déclaration des variables, il y a autant de variable que de pattern à appeler.



Void devant la fonction définit le fait qu'elle ne renvoie pas de valeur. La fonction setup( ) sert à indiquer les options par défaut tel que la définition du cadre vidéo (sa taille), le lancement de la vidéo, de l'enregistrement vidéo et la détection des marqueurs.

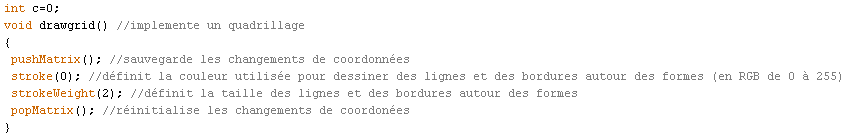


Déclaration des marqueurs, chaque variable est associée à un fichier du pattern, ainsi la variable pourra reconnaître le marker grâce au fichier *.patt* appelé. Par exemple pour la variable *nya\_1* on l'associe au *marker1.patt*, ainsi il peut lire et reconnaître le marker 1.

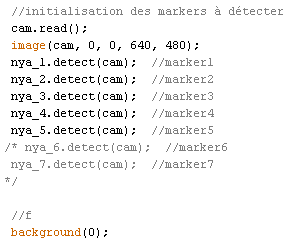
Note : Il faut placer ce fichier marker1.patt dans le répertoire « data »



La fonction drawgrid( ) permet d'établir un quadrillage avec des informations liées au dessin. On peut ainsi définir les couleurs et la taille des lignes et des bordures ainsi que sauvegarder et réinitialiser les changements de coordonnées.



La fonction background permet de changer la couleur du fond de la fenêtre Processing.



On peut alléger, simplifier le programme en définissant une variable contenant les informations pour les zones de texte ( PFont ) de la manière suivante.

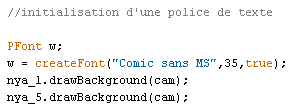
Ainsi on obtient une variable «  w  » qui contient ici les informations suivantes :

-nom de police : Comic sans MS

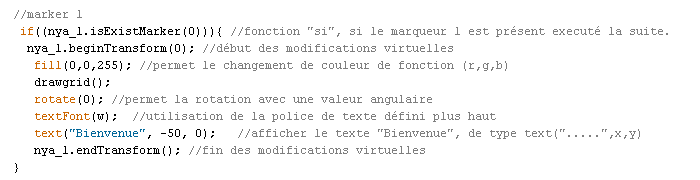
-taille de police : 35

On lui associe ici ces valeurs grâce à la fonction createFont( ).

Il n'y a plus qu'à utiliser la variable comme valeur de la fonction « textFont » pour appeler ces paramètres ci.



Une fonction if( ) contenant les informations à lire par le programme est utilisée pour chaque marker, chacun des markers étant appelés par leur variable ( nya\_1 en l'occurrence pour le marqueur 1). Ici si le marqueur est détecté, le texte « Bienvenue » s'affiche grâce à la fonction text( ) contenant aussi les coordonnées de l'affichage

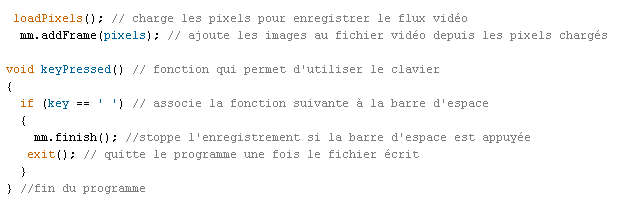


On retrouve donc la fonction if à chaque marker, ici on définit une nouvelle variable de type image grâce à la fonction PImage suivi du nom de la variable ( sti2d dans ce cas) on appelle la variable en l’associant à une image, la fonction loadImage(«...» ) permet de lier la variable à l'image spécifiée (elle doit se trouver dans le même dossier que le fichier, dans le cas contraire il faut spécifier son chemin depuis le fichier processing).

On affiche l'image grâce à la fonction image( ) en lui spécifiant le nom de la variable à utiliser ainsi que sa taille et ses coordonnées.



La partie suivante permet d'enregistrer le flux vidéo, le fichier vidéo étant nommé et définit plus haut dans le programme (lors des déclarations de variables). L'enregistrement se fait au format .ogg. La fonction keyPressed( ) permet d'utiliser une touche pour mettre fin a l'enregistrement et au programme.



Ainsi si on lit le code on remarque la structure suivante (en la simplifiant) :

*Si ( la touche espace est appuyé)*

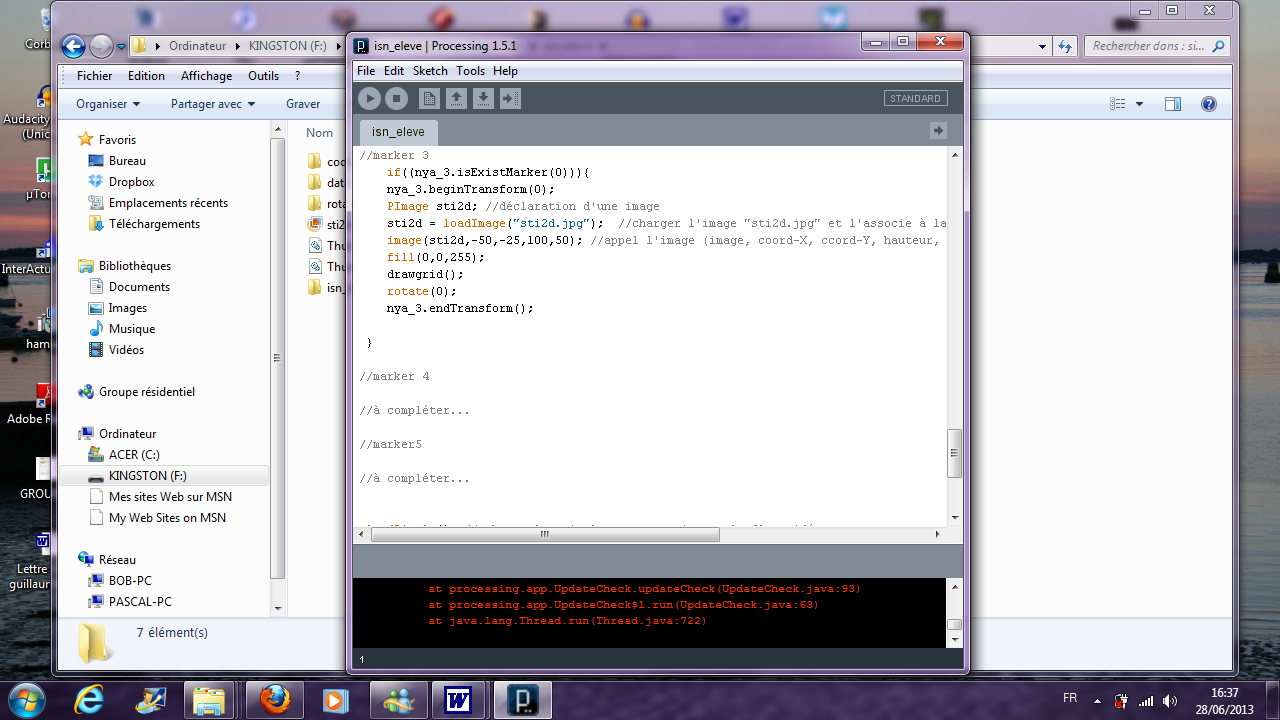
*mettre fin à l'enregistrement*

*fermer le programme*

# 3. Modification du programme – travail demandé

3.1 Charger sur Processing le fichier partiel isn\_élève fourni, analyser la constitution du code utilisé pour l’affichage d’animation à partir des marqueurs 1 à 3.

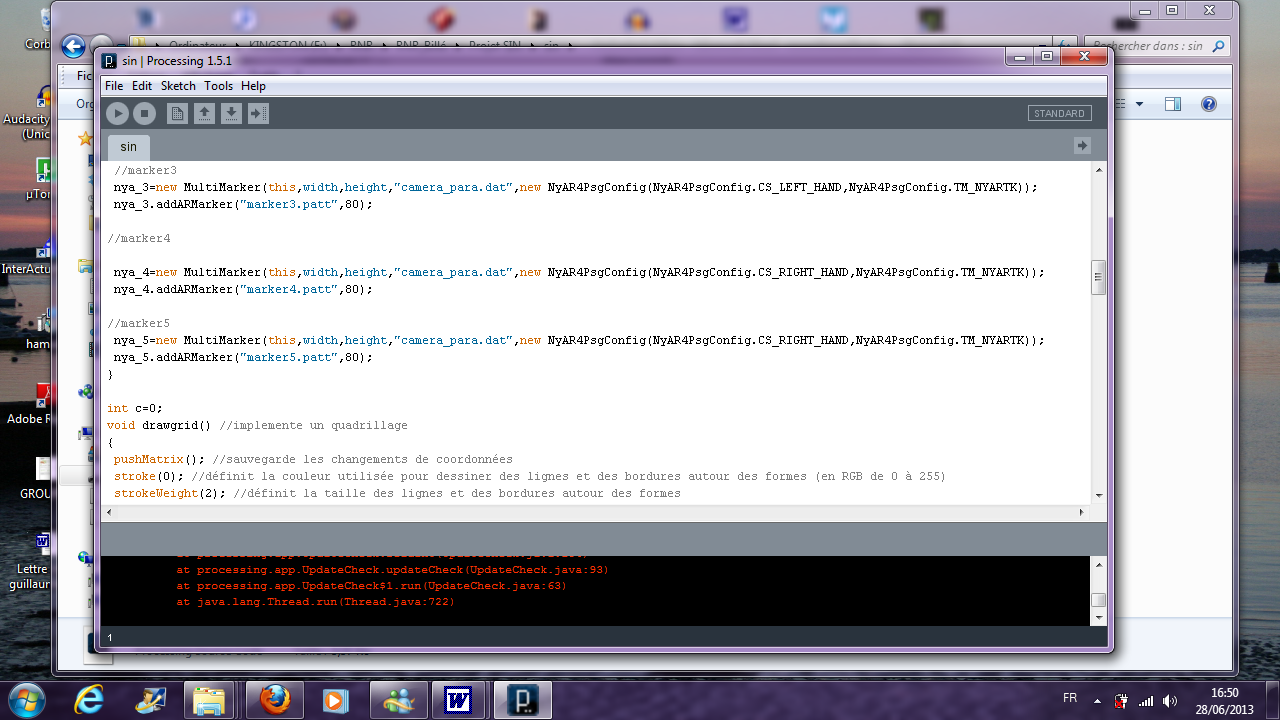
3.2 Compléter ce programme (voir ci-dessous) en écrivant les lignes de code permettant à deux autres marqueurs, que l’on nommera marqueurs 4 et 5, d’afficher un cube de couleur verte non-opaque de côté et une sphère.



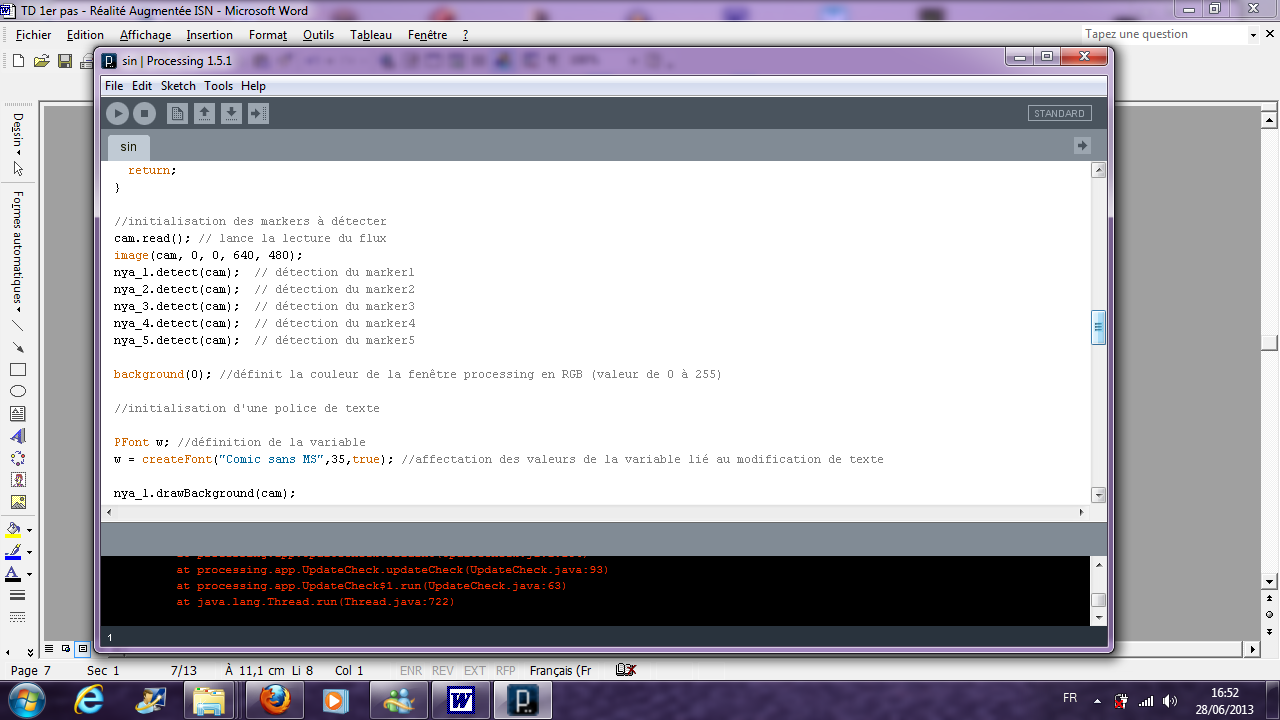
3.3 Enregistrer une vidéo et vérifier le bon fonctionnement de votre programme.

# Correction :

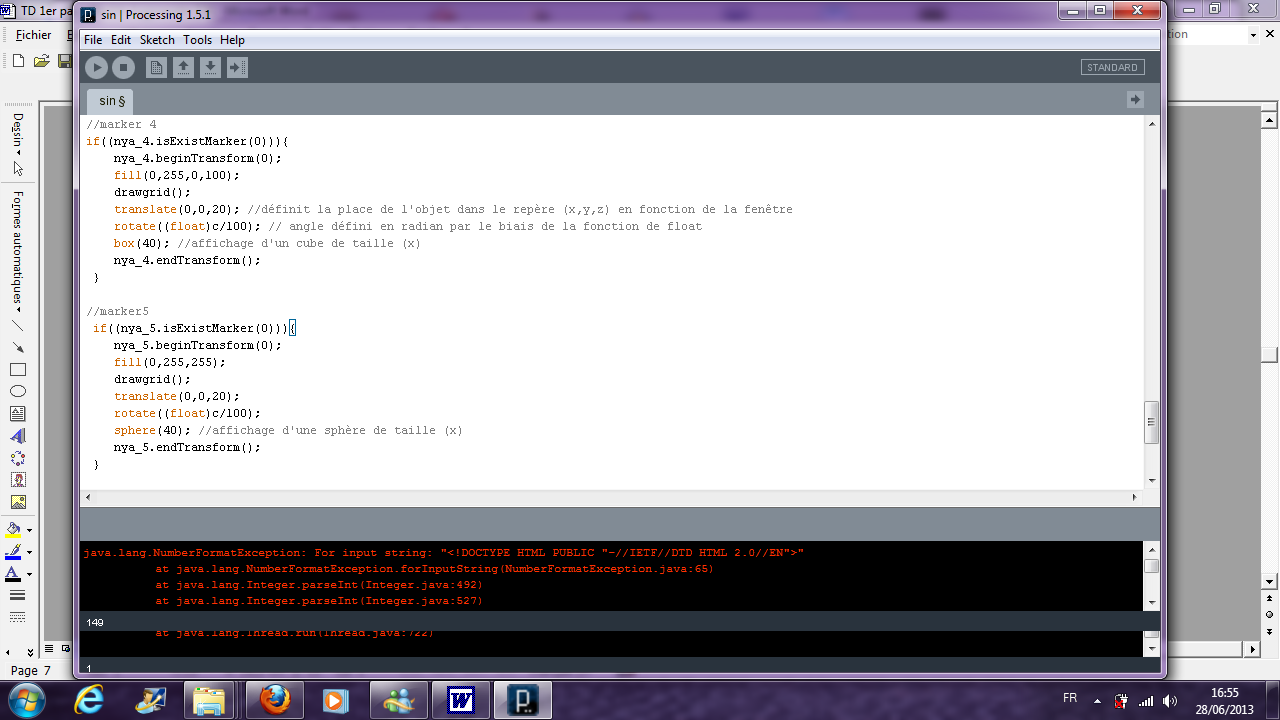
# 



# 



# 

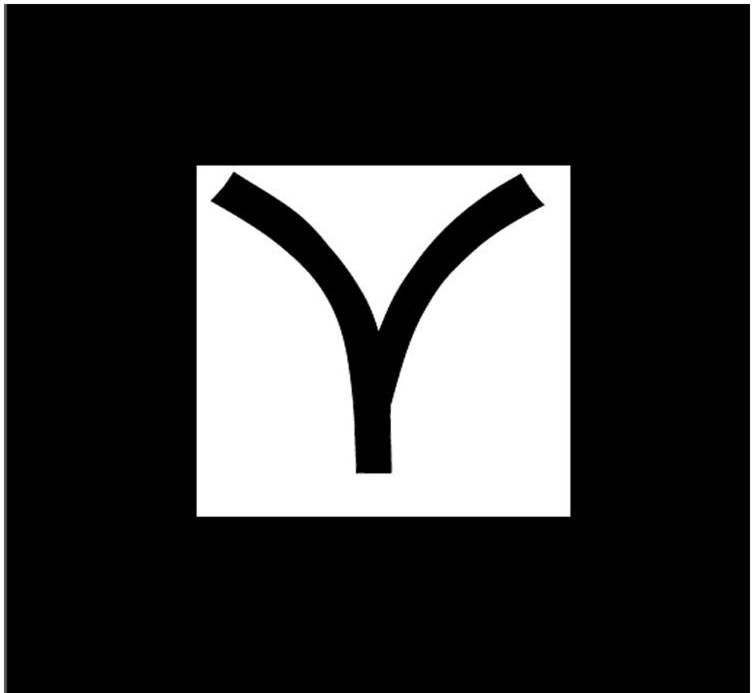


# 4. Les marqueurs utilisés

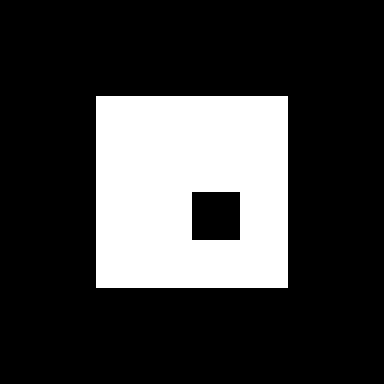
marker 1



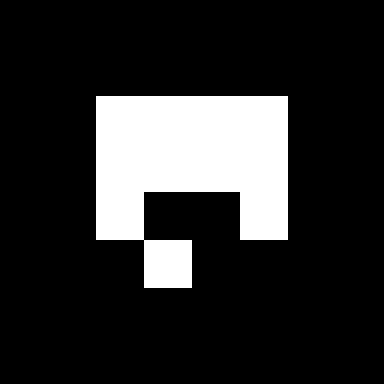
marker 2



marker 3



marker 4



marker 5

