

SAÉ 2.01 – Développement d'une application

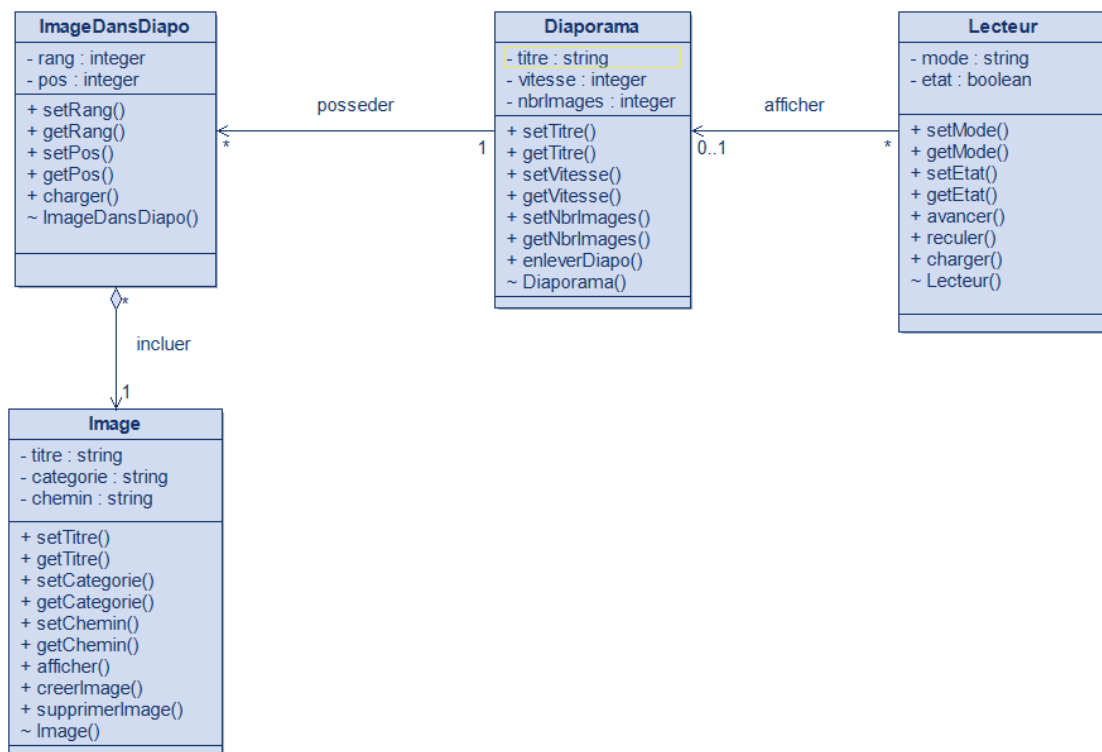
Lecteur de diaporamas – Dossier d'Analyse et conception

<https://github.com/AirG33/LecteurDiaporama>

Version v1 – Projet NON graphique – version OO	2
1. Diagramme de classe (UML)	2
2. Dictionnaire des éléments pour chaque classe	3
3. Capture d'écran du code des différentes classes	6
Lecteur.h	6
Image.h	7
ImageDansDiapo.h	7
Diaporama.h	8
4. Bilan V1	8
Version v2 – Projet NON graphique – version OO	10
Diagramme d'état transition classique	10
Diagramme d'état transition matricielle	11
Lien entre éléments d'interface et fonctionnalités	11

Version v1 – Projet NON graphique – version OO

1. Diagramme de classe (UML)



Remarque sur le diagramme de classe

Nous avons choisis de représenter 4 types de classes :

- La classe **Image** représentant les différentes images stockées et disponibles
- La classe **ImageDansDiapo** représentant les différentes images choisies pour être choisies dans un diaporama
- La classe **Diaporama** qui représente un groupement de photo à afficher
- La classe **Lecteur** qui permet d'afficher selon différents modes le diaporama

Nous avons décidé de créer la classe **ImageDansDiapo** car toutes les images disponibles ne seront pas forcément choisies pour être affichées.

Dans la partie suivante, nous allons voir quelles sont les éléments présents dans les différentes classes, leurs significations et surtout leurs buts (c'est-à-dire pourquoi nous avons décidé de les représenter)

2. Dictionnaire des éléments pour chaque classe

🔗 Attributs et méthodes de chaque classe

Diaporama

Attributs

Nom de l'attribut	Signification	But
titre	Titre du diaporama	Représente le titre du diaporama
vitesse	vitesse de défilement	Représente la vitesse du défilement des images du diaporama
nbrImages	nombres d'images	Représente le nombre d'image présente dans le diaporama

Méthodes

Nom de la méthode	Signification	But
chargerDiapo()	Charger le diaporama	Permet de sélectionner le diaporama souhaité dans une base de données
setTitre()	Définir le titre	Permet de définir le titre du diaporama
getTitre()	Obtenir le titre	Permet d'obtenir le titre du diaporama
setVitesse()	Définir la vitesse de défilement	Permet de définir la vitesse de défilement des images
getVitesse()	Obtenir la vitesse de défilement	Permet d'obtenir la vitesse de défilement des images
setNbrImages()	Définir le nombre d'images	Permet de définir le nombre d'images dans le diaporama
getNbrImages()	Obtenir le nombre d'images	Permet d'obtenir le nombre d'images dans le diaporama
enleverDiapo()	Enlever le diaporama	Permet d'enlever le diaporama voulu
~Diaporama	Destructeur de la classe Diaporama	Permet de détruire l'objet Diaporama

Lecteur

Attributs

Nom de l'attribut	Signification	But
mode	mode du lecteur	Représente le mode de lecture du diaporama
etat	etat du lecteur	Représente l'état du lecteur (vide ou non)

Méthodes

Nom de la méthode	Signification	But
setMode()	Définir le mode de lecture	Permet de définir le mode de lecture du lecteur
getMode()	Obtenir le mode de lecture	Permet d'obtenir le mode de lecture du lecteur
setEtat()	Définir l'état du lecteur	Permet de définir l'état du lecteur
getEtat()	Obtenir l'état du lecteur	Permet d'obtenir l'état du lecteur
avancer()	Avancer	Permet de passer à l'objet suivant dans le lecteur
reculer()	Reculer	Permet de passer à l'objet précédent dans le lecteur
charger()	Charger le diaporama	Permet de charger le diaporama dans le lecteur

ImageDansDiapo

Attributs

Nom de l'attribut	Signification	But
rang	rang de l'image	Représente le rang de l'image dans le diapo
pos	position de l'image	Représente la position initiale de l'image

Méthodes

Nom de la méthode	Signification	But
setRang()	Définir le rang	Permet de définir le rang de l'image dans le diapo
getRang()	Obtenir le rang	Permet d'obtenir le rang de l'image dans le diapo
setPos()	Définir la position	Permet de définir la position initiale de l'image
getPos()	Obtenir la position	Permet d'obtenir la position initiale de l'image
charger()	charger l'image	Permet de charger l'image dans le diaporama
~ImageDansDiapo()	Destructeur de la classe ImageDansDiapo	Permet de détruire l'objet ImageDansDiapo

Image

Attributs

Nom de l'attribut	Signification	But
titre	Intitulé / titre	Représente le titre / intitulé de l'image
categorie	Catégorie	Représente la catégorie de l'image
cheminAcces	Chemin d'accès	Représente le chemin d'accès à l'image

Méthodes

Nom de la méthode	Signification	But
setTitre()	Définir le titre	Permet de définir le titre de l'image
getTitre()	Obtenir le titre	Permet d'obtenir le titre de l'image
setCategorie()	Définir la catégorie	Permet de définir la catégorie de l'image
getCategorie()	Obtenir la catégorie	Permet d'obtenir la catégorie de l'image
setCheminAcces()	Définir le chemin d'accès	Permet de définir le chemin d'accès à l'image
getCheminAcces()	Obtenir le chemin d'accès	Permet d'obtenir le chemin d'accès à l'image
afficher()	Afficher l'image	Permet d'afficher l'image voulue
creerImage()	Créer une image	Permet de créer une image
supprimerImage()	Supprimer une image	Permet de supprimer une image
~Image()	Destructeur de la classe Image	Permet de détruire l'objet Image

3. Capture d'écran du code des différentes classes

Lecteur.h

```
1  #ifndef LECTEUR_H
2  #define LECTEUR_H
3  #include <iostream>
4  using namespace std;
5  #include "diaporama.h"
6
7
8  class Lecteur
9  {
10 private:
11     unsigned int nbrImages;
12     string mode;
13     bool etat;
14 public:
15     Lecteur();
16     ~Lecteur();
17     void setNbImages(unsigned int nbrImages);
18     unsigned int getNbImages();
19     void setMode(string mode);
20     string getMode();
21     void setEtat(bool etat);
22     string getEtat();
23     void avancer();
24     void reculer();
25     void charger(Diaporama diapo);
26
27
28
29 };
30
31 #endif // LECTEUR_H
32
```

Image.h

```
1  #ifndef IMAGE_H
2  #define IMAGE_H
3  #include <iostream>
4  using namespace std;
5
6  // Module de manipulation d'éléments de type Image
7
8  class Image
9  {
10 private:
11     string titre;           // intitulé de l'image
12     string categorie;       // catégorie de l'image (personne, animal, objet)
13     string chemin;
14 public:
15
16     Image();
17     ~Image();
18     void setTitre(string titre);
19     string getTitre();
20     void setCategorie(string categorie);
21     string getCategorie();
22     void setChemin(string chemin);
23     string getChemin();
24     void afficher();
25     void creerImage ();
26     void supprimerImage();
27 };
28
29 #endif // IMAGE_H
30
```

ImageDansDiapo.h

```
1  #ifndef IMAGEDANSDIAP0_H
2  #define IMAGEDANSDIAP0_H
3  #include <iostream>
4  using namespace std;
5  #include "image.h"
6
7  class ImageDansDiapo
8  {
9  private:
10     int rang; // rang de l'image dans le diaporama
11               // = ordre d'affichage choisi par l'utilisateur lors de la création du diaporama
12     int pos; // rang de l'image dans le tableau d'images (vector<Images>)
13               // = ordre de chargement initial des images dans la table des images
14
15 public:
16
17     ImageDansDiapo();
18     ~ImageDansDiapo();
19     void setRang(int rang);
20     int getRang();
21     void setPos(int pos);
22     int getPos();
23     void charger(Image Image);
24 };
25
26 #endif // IMAGEDANSDIAP0_H
27
```

Diaporama.h

```
1  #ifndef DIAPORAMA_H
2  #define DIAPORAMA_H
3  #include <iostream>
4  using namespace std;
5
6  class Diaporama
7  {
8  private:
9      string titre;
10     unsigned int vitesse;
11     unsigned int nbrImages;
12
13
14 public:
15
16     Diaporama();
17     ~Diaporama();
18     void setTitre(string titre);
19     string getTitre();
20     void setVitesse(unsigned int vitesse);
21     unsigned int getVitesse();
22     void setNbrImages(unsigned int nbrImages);
23     unsigned int getNbrImages();
24     void enleverDiapo();
25 };
26
27 #endif // DIAPORAMA_H
28
```

4. Bilan V1

Dépôt Git où trouver le projet complet:

<https://github.com/AirG33/LecteurDiaporama.git>

Nous avons fait pour le rendu du 01/05/2024 la version V1 complète.

environ 30 commits ont été effectués sur ce repository (certains bien plus utiles que d'autres)

Temps global de travail:

Nous avons travaillé durant le peu de séances de SAE ce qui représente 4h30 heures de projet au sein de l'iut, puis depuis chez nous, nous avons travaillé environ 6 heures en groupe ou à distance hors IUT. Ce qui ferait un total d'environ 10 heures de travail pour cette version du projet.

Difficultés majeures:

Nos principales difficultés ont été rencontrées lors de la compréhension du sujet et de la réalisation du diagramme des classes qui à été repris de nombreuses fois avant d'obtenir une version convenable à nos yeux. Nous avons ressenti pour nous, un léger manque d'accompagnement qui nous a ralenti et nous a toujours demandé de nous remettre en questions sur nos choix, ce qui explique le retard déjà causé lors de cette première version.

Points positifs / négatifs de l'activité:

Les points positifs ont été le choix du thème qui est intéressant et donne envie de réaliser le projet.

Pour ce qui est des points négatifs nous pourrions citer le manque d'accompagnement par certains enseignants lors de séances encadrés ainsi que le formalisme de certaines demandes qui n'était pas forcément clair pour nous. Ainsi que le retard de commencement de la SAE qui n'a pas été modifié par la suite.

Il y a eu aussi pour notre part un réel manque de temps pour pouvoir atteindre tous les objectifs de manière rigoureuses. Ce qui a donc fait que nous ne sommes pas arrivés à une version V1 parfaite.

Tableau de la répartition horaire de la SAE entre les membres:

	Félix Autant	Rémi Gentil	Florian Bonneau
Documentation	1h	3h	2h
V1	4h	2h	2h

Version v2 – Projet NON graphique – version OO

Diagramme d'état transition classique

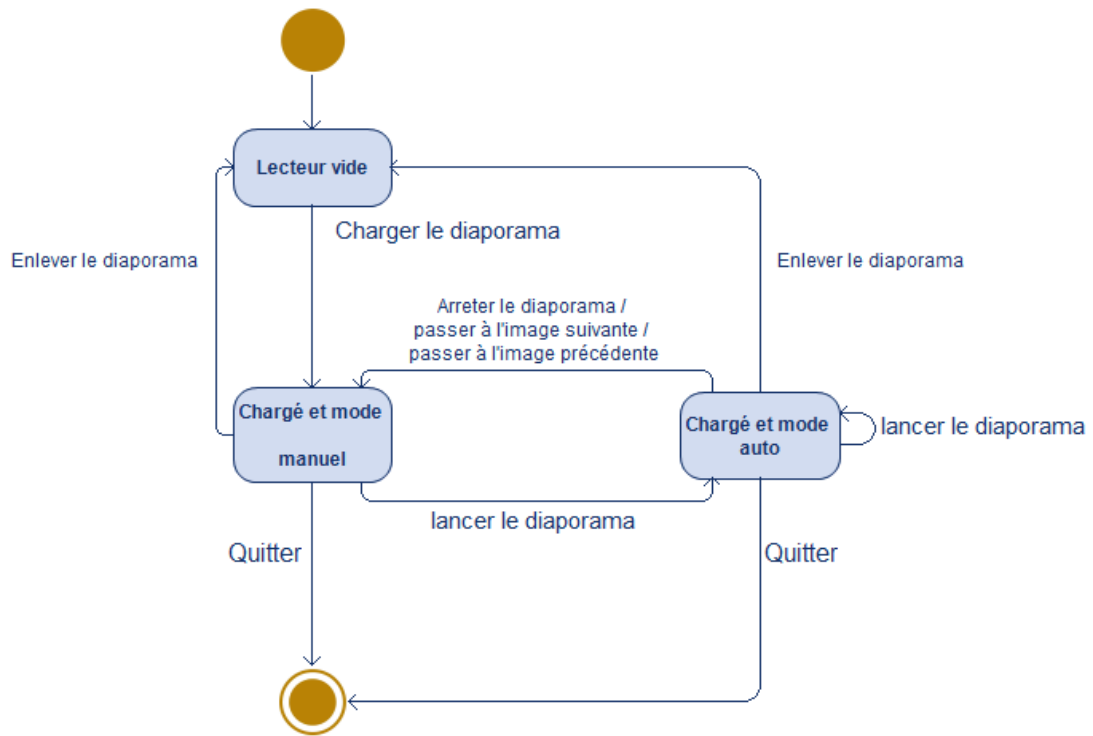



Diagramme d'état transition matricielle

Éléments d'interface déclencheurs des événements	acCharger	bLancerDiapo	bArreterDiapo	bSuivant	bPrecedent	acVider	acQuitter
Evénements →  Etats ↓	Charger le diaporama	Lancer le diaporama	Arrêter le diaporama	Passer à l'image suivante	Passer à l'image précédente	Enlever le diaporama	Quitter
Lecteur vide	Chargé et mode manuel	x	x	x	x	x	x
Chargé et mode manuel	x	Chargé et mode auto	x	Chargé et mode manuel	Chargé et mode manuel	Lecteur vide	x
Chargé et mode auto	x	x	Chargé et mode manuel	Chargé et mode manuel	Chargé et mode manuel	Lecteur vide	x

Lien entre éléments d'interface et fonctionnalités

acCharger : action permettant de charger le diaporama dans le lecteur

acVider : action permettant de vider le diaporama

acAproposde : action permettant d'ouvrir une boîte de Message indiquant les auteurs de l'application et la version de l'application

acQuitter : action permettant de quitter l'application

bLancerDiapo : Bouton permettant de lancer le diaporama en mode automatique

bArreterDiapo : Bouton permettant d'arrêter le diaporama et par conséquent de passer en mode manuel

bSuivant : Bouton permettant de passer à l'image suivante en mode manuel

bPrecedent : Bouton permettant de passer à l'image précédente en mode manuel

ITitreDiapo : label permettant d'afficher le titre du diapo

ITitreImage : label permettant d'afficher le titre de l'image

ICategoriImage : label permettant d'afficher la catégorie de l'image

IRangImage : label permettant d'afficher le rang de l'image

ITotalNblImages : label permettant d'afficher le nombre total d'images

IMode : label permettant d'afficher le mode actuel de lecture (auto ou manuel)

Version 3 - Projet Qt graphique, créé par duplication de la v2

Dans cette version, nous avons pour objectif de créer la partie graphique de l'application et d'implémenter les différentes fonctionnalités.

Nous avons eu à afficher dans l'interface graphique, des images provenant d'une ressource annexe et non d'une base de données pour le moment.

Les fonctionnalités qui ont été développés et implémenté dans le projet sont :

- Fonctionnement du lecteur en mode Manuel uniquement
- L'arrêt de l'application en accédant à l'option Quitter dans le menu Fichier
- La possibilité de se renseigner sur l'application en ouvrant une Boîte de Message via l'option A propos de ... dans le menu Aide. Cette Boîte de Message donne des informations sur :
 - la version de l'application
 - la date de création
 - les auteurs