Saé 2.01 – Développement d'une application

Lecteur de diaporamas – Dossier d'Analyse et conception

1. Compléments de spécifications externes.

1.1 Maquettes

Des maquettes ont été réalisées pour clarifier l'organisation des différents éléments et des différentes informations qui doivent figurer dans les écrans du diaporama :

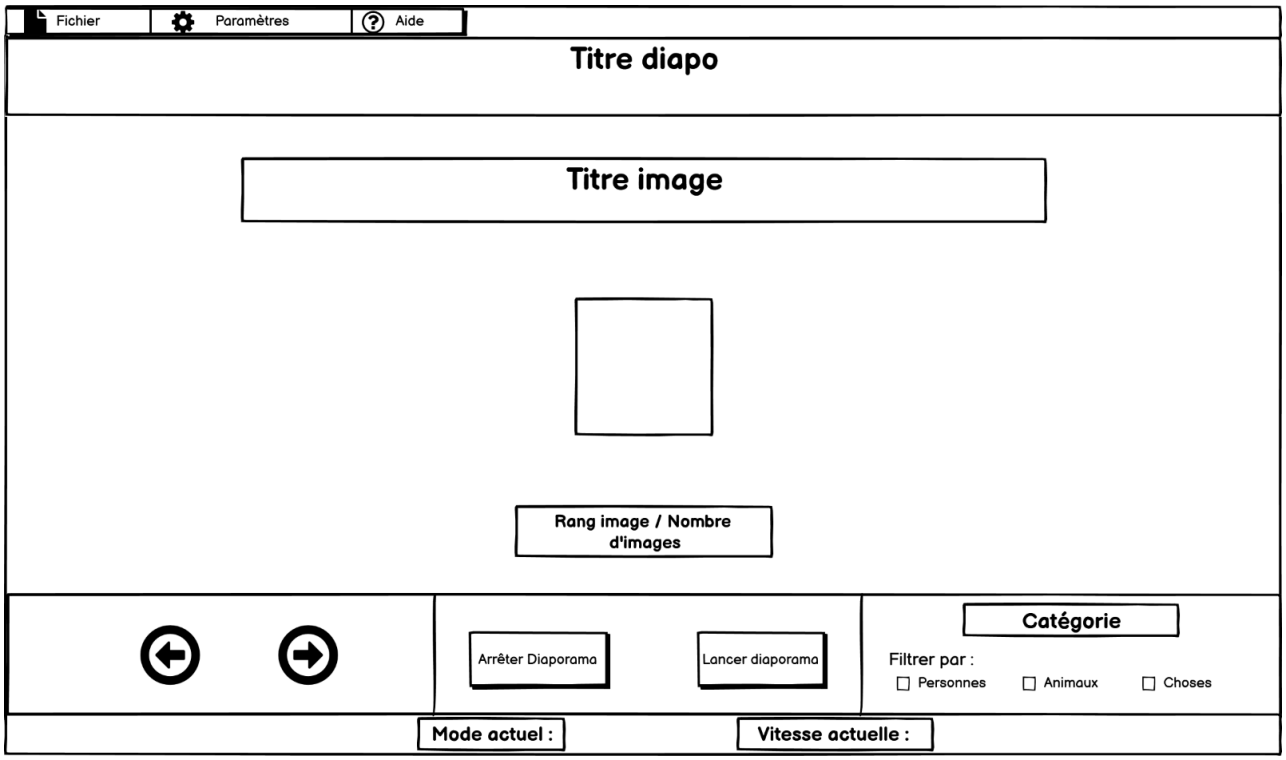


Figure 1 - Maquette Principale

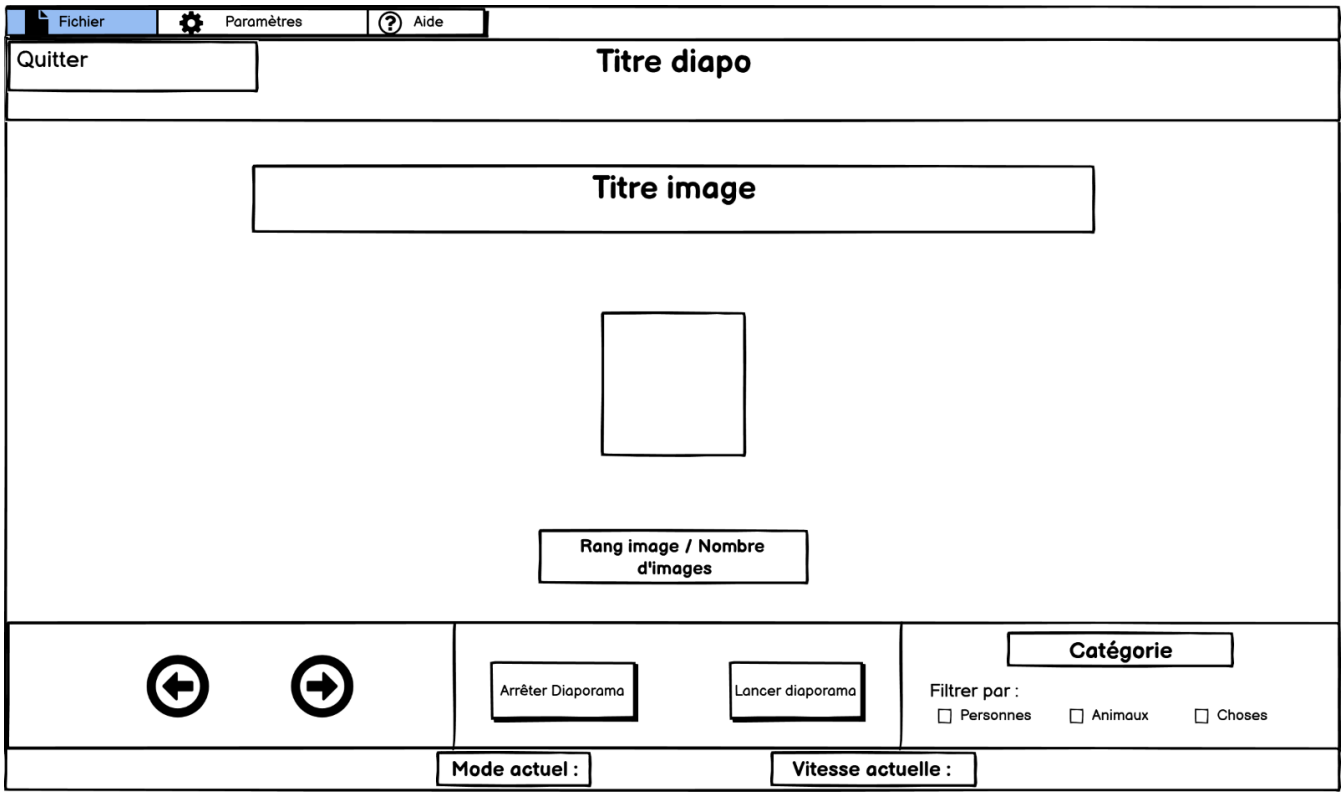


Figure 2 - Maquette Survol Fichier

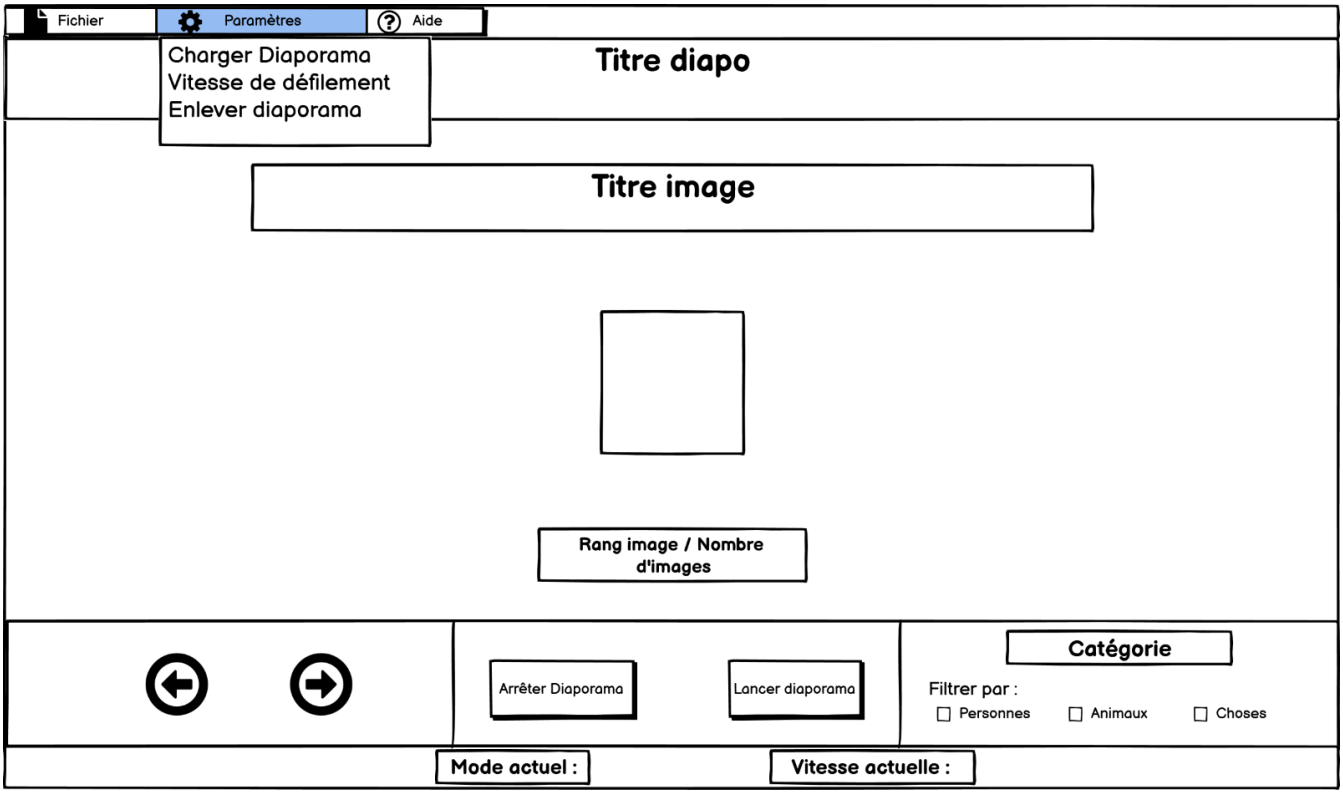


Figure 3 - Maquette Survol Paramètres

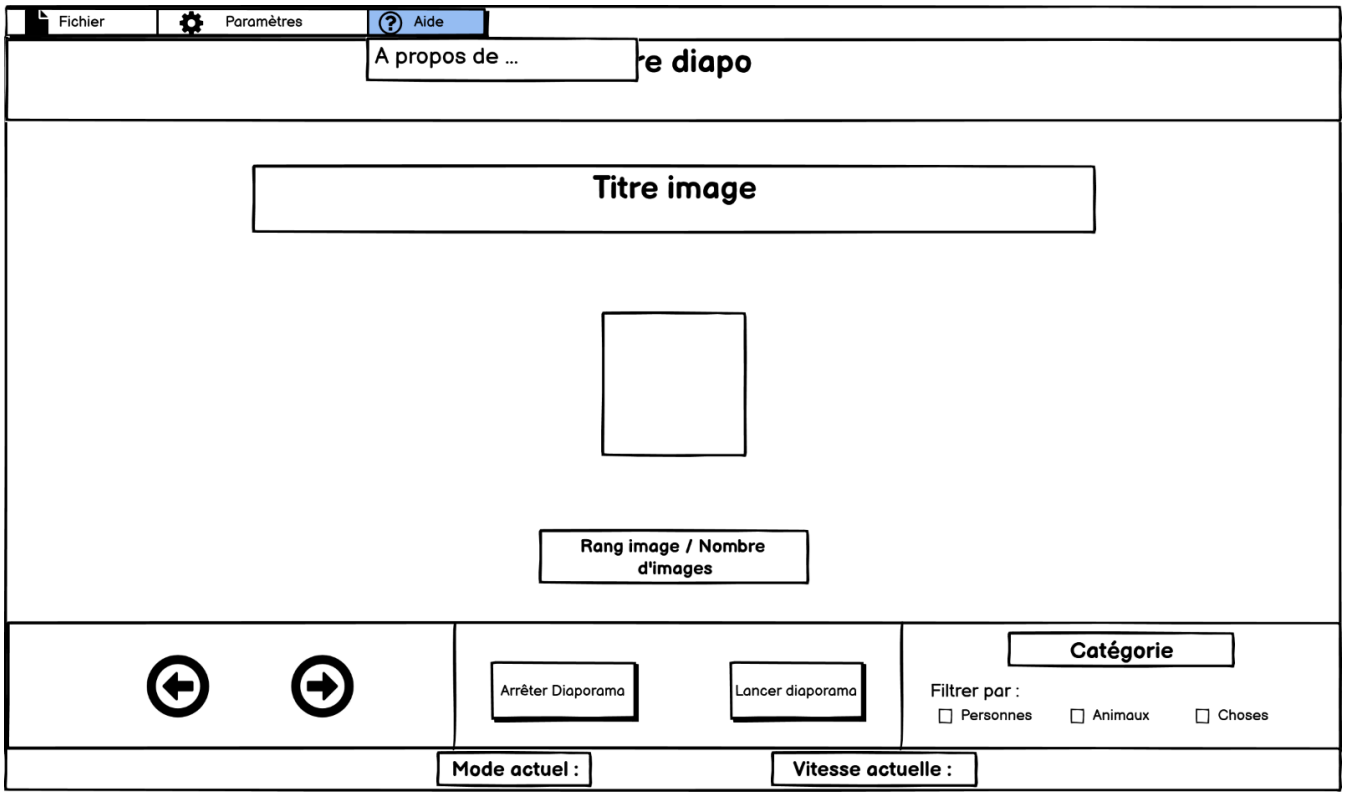


Figure 4 - Maquette Survol Aide

1.2 Diagramme des cas d'utilisations

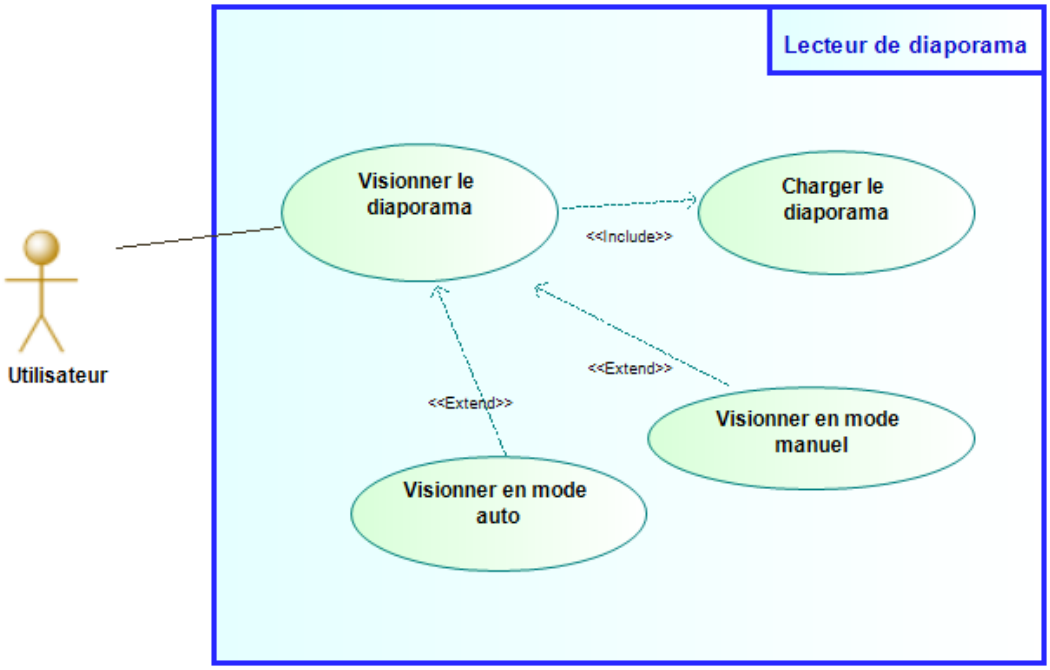


Figure 5 - Diagramme des cas d'utilisations

En utilisant le lecteur de diaporama, l'utilisateur cherche à visionner le diaporama qu'il souhaite. Pour cela, il devra obligatoirement le charger et aura la possibilité de le visionner en mode manuel, c'est-à-dire en utilisant les boutons, ou en mode auto pour avoir un défilement automatique en fonction d'une vitesse exprimée en secondes qu'il pourra définir.

2. Scénarios

Titre : L'utilisateur lance son diaporama

Acteurs : Utilisateur

Résumé : Un utilisateur souhaite lire son diaporama

Fiche : Description Réelle

Métadonnées :

Création : 16/05/2023

Responsable : AMBROISE Axel

Modifiée le : 16/05/2023 Version : 1

Enchainements

Préconditions :

1. L'utilisateur a lancé l'application.

Postconditions :

1. Le diaporama est lu

Enchaînement Nominal :

Acteur :	Système :	Maquettes
1. L'utilisateur va dans les paramètres		Figure 1 - Maquette Principale
	2. Le système affiche le menu Paramètre.	
3. L'utilisateur demande à charger son diaporama		

	4. Le système affiche le menu de choix du diaporama.	
5. L'utilisateur sélectionne son diaporama.		
	6. Le système récupère le diaporama et l'affiche.	Figure 1 - Maquette Principale
7. L'utilisateur navigue à l'aide des boutons disponibles.		

Enchaînement Alternatif :

L'utilisateur lance le diaporama en mode automatique puis change la vitesse de défilement.

Acteur :	**Système :**	**Maquettes**
7. Point de départ à l'élément 7 de l'enchainement nominal L'utilisateur appuie sur le bouton de lancement de diaporama		Figure 1 - Maquette Principale
	8. Le système passe du mode manuel au mode automatique	
9. L'utilisateur va dans les paramètres		Figure 3 - Maquette Survol Paramètres
	10. Le système affiche le menu Paramètres.	
11. L'utilisateur sélectionne l'option de changement de vitesse de défilement.		
	12. Le système affiche une boite de dialogue pour sélectionner la vitesse de défilement voulue.	Pas de maquettes pour ces éléments.
13. L'utilisateur entre la vitesse qu'il souhaite puis valide.		
	14. Le système modifie la vitesse de défilement.	

Enchaînement Alternatif :

L'utilisateur enlève son diaporama. On suppose donc qu'il y a déjà un diaporama de chargé dans le lecteur.

Acteur :	Système :	Maquettes
1. L'utilisateur va dans les paramètres.		Figure 3 - Maquette Survol Paramètres
	2. Le système affiche le menu Paramètres.	
3. L'utilisateur demande à enlever son diaporama		
	4. Le système vide le lecteur de diaporama.	
	5. Le système affiche l'écran principal.	Figure 1 - Maquette Principale

3. Diagramme de classe (UML)

(a) Le diagramme de classes UML se focalise sur les classes **métier**, cad celles décrivant les éléments structurants de l'application, indépendamment des éléments d'interface.

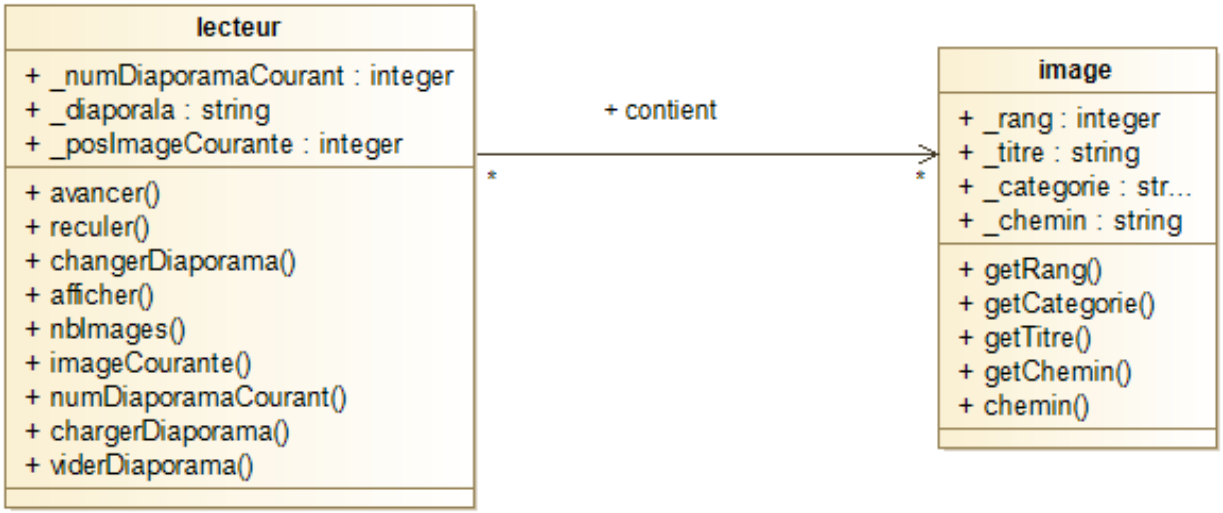


Figure 6 - Diagramme de classe (UML)

(b) Dictionnaire des éléments pour chaque classe

Classe Image			
Nom Attribut	**Signification**	**Type**	**Exemple**
_rang	Le rang de l'image au sein du diaporama auquel l'image est associée	unsigned int	4
_titre	L'intitulé de l'image	string	"Mickey"
_categorie	La catégorie de l'image (personne, animal, objet)	string	"Animal"
_chemin	Le chemin complet vers le dossier où se trouve l'image	string	"C:\\cartesDisney\\carteDisney1.gif"

Tableau 1 : Dictionnaire des éléments - Classe Image

Classe Lecteur			
Nom Attribut	**Signification**	**Type**	**Exemple**
_numDiaporamaCourant	Le numéro du diaporama courant, par défaut 0	unsigned int	1
_diaporama	Le pointeur vers les images du diaporama	Diaporama	monDiapo
_posImageCourante	La position, dans le diaporama, de l'image courante. Indéfini quand diaporama vide. Démarre à 0 quand diaporama non vide	Unsigned int	6

Tableau 2 : Dictionnaire des éléments - Classe Lecteur

(c) Dictionnaire des méthodes : vous pouvez fournir directement le fichier entête de chaque classe.

Exemple (classe lecteur de la version Console):

```
#ifndef LECTEUR_H
#define LECTEUR_H
#include "image.h"
#include <vector>

typedef vector<Image*> Diaporama;   // Structure de données contenant les infos sur les images

class Lecteur
{
public:
    Lecteur();
    void avancer();             // incrémente _posImageCourante, modulo nbImages()
    void reculer();             // décrémente _posImageCourante, modulo nbImages()
    void changerDiaporama(unsigned int pNumDiaporama);    // permet de choisir un diaporama, 0 si aucun diaporama souhaité
    void afficher();            // affiche les informations sur lecteur-diaporama et image courante
    unsigned int nbImages();    // affiche la taille de _diaporama
    Image* imageCourante();     // retourne le pointeur vers l'image courante
    unsigned int numDiaporamaCourant();

private:
    unsigned _numDiaporamaCourant;   // numéro du diaporama courant, par défaut 0
    Diaporama _diaporama;            // pointeurs vers les images du diaporama
    unsigned int _posImageCourante;  /* position, dans le diaporama,
                                        de l'image courante.
                                        Indéfini quand diaporama vide.
                                        Démarre à 0 quand diaporama non vide */
private:
    void chargerDiaporama();    // charge dans _diaporama les images du _numDiaporamaCourant
    void viderDiaporama();      // vide _diaporama de tous ses objets image et les delete
};

#endif // LECTEUR_H
```

Figure 7 : Schéma de classes = Classe Lecteur

```
#ifndef IMAGE_H
#define IMAGE_H
#include <iostream>
using namespace std;

class Image
{
public:
    Image(unsigned int pRang=0,
              string pCategorie="", string pTitre="", string pChemin = "");
    unsigned int getRang();
    string getCategorie();
    string getTitre();
    string getChemin();
    void afficher();           // affiche tous les champs de l'image

private:
    unsigned int _rang;         /* rang de l'image au sein du diaporama
                                   auquel l'image est associée */
    string _titre;              // intitulé de l'image
    string _categorie;          // catégorie de l'image (personne, animal, objet)
    string _chemin;             // chemin complet vers le dossier où se trouve l'image
};

#endif // IMAGE_H
```

Figure 8 – Schéma de classes – Classe Image

(d) Remarques concernant le schéma de classes

1. On ne s'intéresse qu'aux attributs et méthodes métier. Notamment, on ne met pas, pour l'instant, ce qui relève de l'affichage car ce sont d'autres objets du programme (widgets) qui se chargeront de l'affichage. Par contre, on n'oublie pas les méthodes getXXX(), qui permettront aux objets métier de communiquer leur valeur aux objets graphiques pour que ceux-ci s'affichent.
2. On n'a mis ni le constructeur ni le destructeur, pour alléger le schéma.
3. D'autres attributs et méthodes pourront venir ultérieurement compléter cette première vision ANALYTIQUE de l'application. Il s'agira des attributs et méthodes dits DE CONCEPTION nécessaires au développement de l'application.

Version v0 – Version console seule

4. Implémentation et tests

4.1 Implémentation

Liste et rôle des fichiers de cette version :

lecteur.h	Spécification de la classe Lecteur
lecteur.cpp	Corps de la classe Lecteur
image.h	Spécification de la classe Image
image.cpp	Corps de la classe Image
main.cpp	Teste les méthodes de la classe Lecteur

4.2 Test

Test avec le programme fourni main.cpp
Les tests réalisés pour la v-0 figurent ici sous forme d'images et non sous forme de tableau insérés dans le fichier pour la lisibilité du document.

	Testeur : M.Guiheneuf		**Date :** 10/05/2023		
	Element Testé : LecteurDiaporama : v-0		**Version :** 1.0		
Classe	**Description**	**Valeur(s) en entrée**	**Résultat(s) attendu(s)**	**Résultats Finaux**	**Conforme**
Valide n°1	Sélection du diaporama : Statut du lecteur avant la sélection, message de sélection, nombre d'images chargées et affichage de l'image courante	pNumDiaporama = 1	Lecteur vide Diaporama num. 1 selectionne.	Lecteur vide Diaporama num. 1 selectionne.	
Valide n°2	Test avancer () : 4 fois ==> 1ère fois	-	avancer() : Diaporama num. 1 Image courante : image(rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)	avancer() : Diaporama num. 1 Image courante : image(rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)	
Valide n°3	Test avancer () : 4 fois ==> 2ème fois	-	avancer() : Diaporama num. 1 Image courante : image(rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)	avancer() : Diaporama num. 1 Image courante : image(rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)	
Valide n°4	Test avancer () : 4 fois ==> 3ème fois	-	avancer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)	avancer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)	
Valide n°5	Test avancer () : 4 fois ==> 4ème fois	-	avancer() : Diaporama num. 1 Image courante : image(rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)	avancer() : Diaporama num. 1 Image courante : image(rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)	
Valide n°6	Test reculer () : 5 fois ==> 1ère fois	-	reculer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)	reculer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)	
Valide n°7	Test reculer () : 5 fois ==> 2ème fois	-	reculer() : Diaporama num. 1 Image courante : image(rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)	reculer() : Diaporama num. 1 Image courante : image(rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)	

Figure 9 - Test v-0 1

Valide n°8	Test reculer () : 5 fois ==> 3ème fois	-	reculer() : Diaporama num. 1 Image courante : image(rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)	reculer() : Diaporama num. 1 Image courante : image(rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)
Valide n°9	Test reculer () : 5 fois ==> 4ème fois	-	reculer() : Diaporama num. 1 Image courante : image(rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)	reculer() : Diaporama num. 1 Image courante : image(rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
Valide n°10	Test reculer () : 5 fois ==> 5ème fois	-	reculer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)	reculer() : Diaporama num. 1 Image courante : image(rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)
Valide n°11	Enlever le diaporama : Choix du diaporama 0, nombre d'images dans le lecteur et statut du lecteur	pNumDiaporama = 0	Enlever le diaporama courant = Choisir diaporama 0 0 images restantes dans le diaporama. Lecteur vide	Enlever le diaporama courant = Choisir diaporama 0 0 images restantes dans le diaporama. Lecteur vide

Figure 10 - Test v-0 2

Version v1 – projet Graphique seul

5. Éléments d'interface

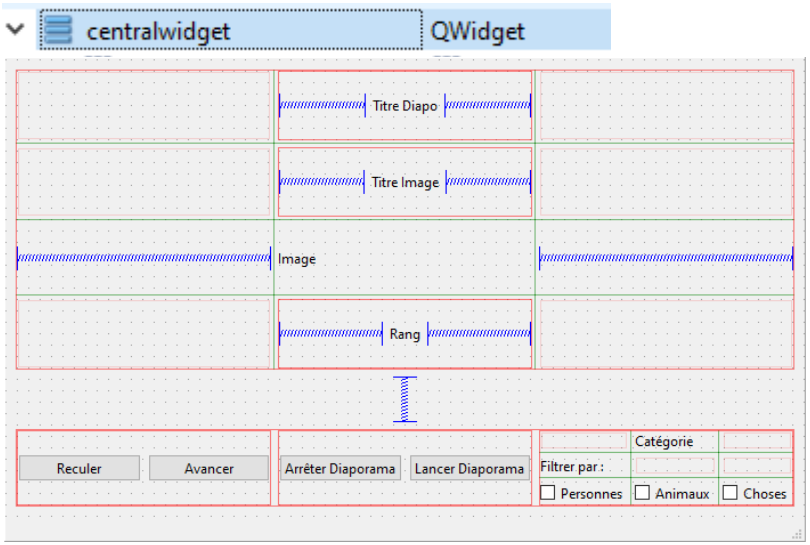


Figure 11 - Eléments d'interfaces - CentralWidget

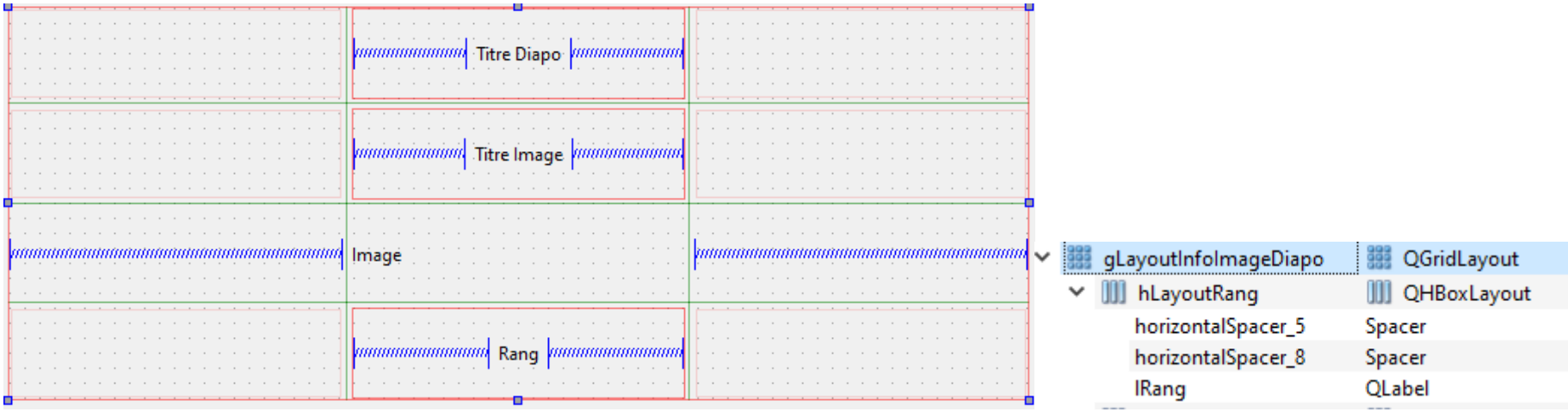


Figure 12 - Eléments d'interfaces - gLayoutInfoImageDiapo

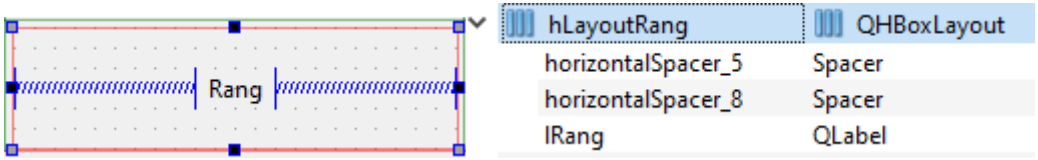


Figure 13 - Eléments d'interfaces - hLayoutRang

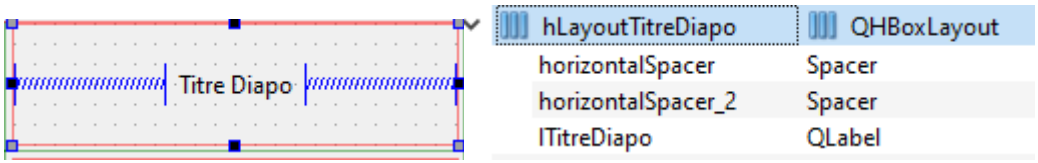


Figure 14 - Eléments d'interfaces - hLayoutTitreDiapo

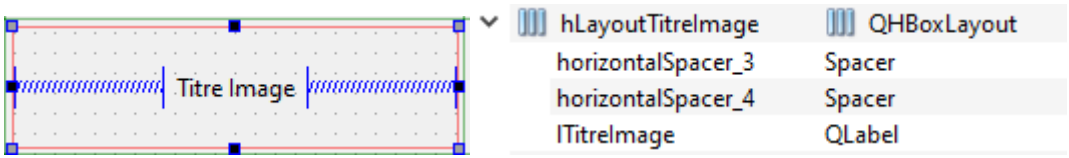


Figure 15 - Eléments d'interfaces - hLayoutTitreImage

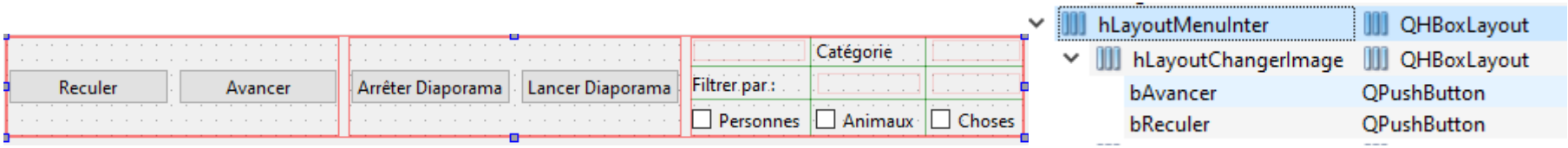


Figure 16 - Eléments d'interfaces - hLayoutMenuInter

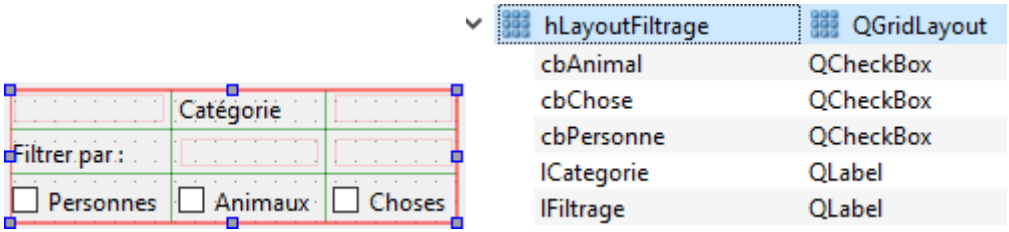


Figure 17 - Eléments d'interfaces - gLayoutFiltrage

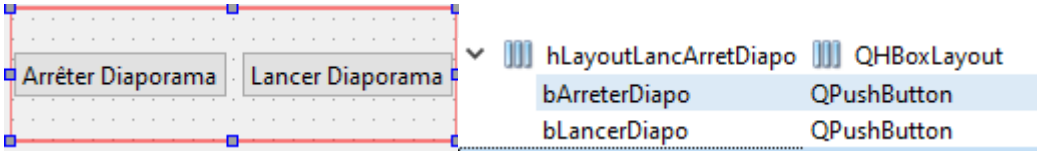
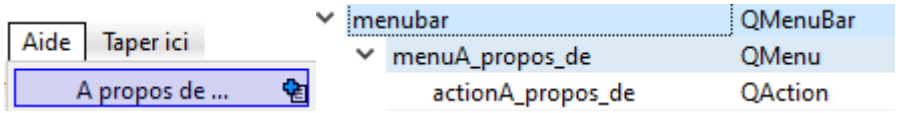
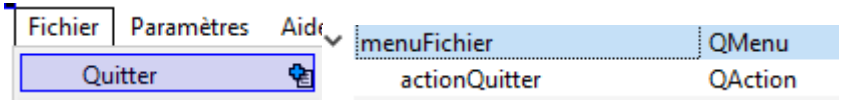


Figure 18 - Eléments d'interfaces - hLayoutLancArretDiapo

menubar :



menuFichier :



menuParametres :

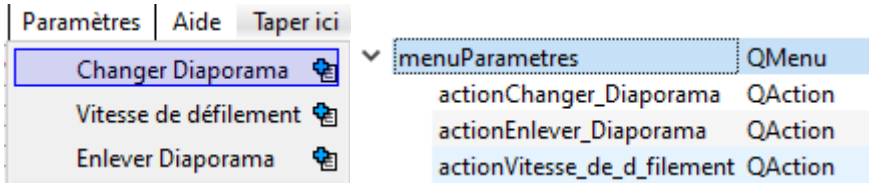


Figure 19 - Eléments d'interfaces - menuBar

Implémentation et tests

6.1 Implémentation

Liste et rôle des fichiers de cette version :

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas
lecteurVue.cpp	Corps de la classe LecteurVue
lecteurvue.ui	Fichier du dessin de l'interface réalisé par QtDesigner
main.cpp	Teste les méthodes de la classe Lecteur

Remarques sur l'implémentation :

Les premiers boutons de l'interface graphique ont été connectés aux méthodes correspondantes :

- bAvancer connecté avec avancer()
- bReculer connecté avec reculer()
- bLancerDiapo et bArreterDiapo respectivement avec lancerDiapo() et arrêterDiapo()
- les boutons de filtrage cbPersonne, cbAnimal et cbObjet tous les trois connectés avec filtrerImages() qui sera développer plus tard.

Pour l'instant, ces méthodes sont constituées de qDebug(), affichant un message confirmant qu'une action a été effectuée.

6.2 Test

Les tests réalisés consistent à vérifier que lors de la mise en marche d'un bouton, la méthode adaptée est lancée.

Testeur : M.Guiheneuf **Date :** 17/05/2023

Element Testé : LecteurDiaporama : v-1 **Version :** 1.0

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	Résultats Finaux	Conforme
Valide n°1	Emploi du bouton bAvancer	-	"L'utilisateur avance"	"L'utilisateur avance"	
Valide n°2	Emploi du bouton bReculer	-	"L'utilisateur recule"	"L'utilisateur recule"	
Valide n°3	Emploi du bouton bLancerDiapo	-	"L'utilisateur lance le diaporama"	"L'utilisateur lance le diaporama"	
Valide n°4	Emploi du bouton bArreterDiapo	-	"L'utilisateur arrête le diaporama"	"L'utilisateur arrête le diaporama"	
Valide n°5	Emploi de la checkbox cbPersonne	-	"L'utilisateur filtre les images"	"L'utilisateur filtre les images"	
Valide n°6	Emploi de la checkbox cbAnimal	-	"L'utilisateur filtre les images"	"L'utilisateur filtre les images"	
Valide n°7	Emploi de la checkbox cbObjet	-	"L'utilisateur filtre les images"	"L'utilisateur filtre les images"	

Figure 20 - Test v-1

Version v2 –

Diagramme de classes (UML)

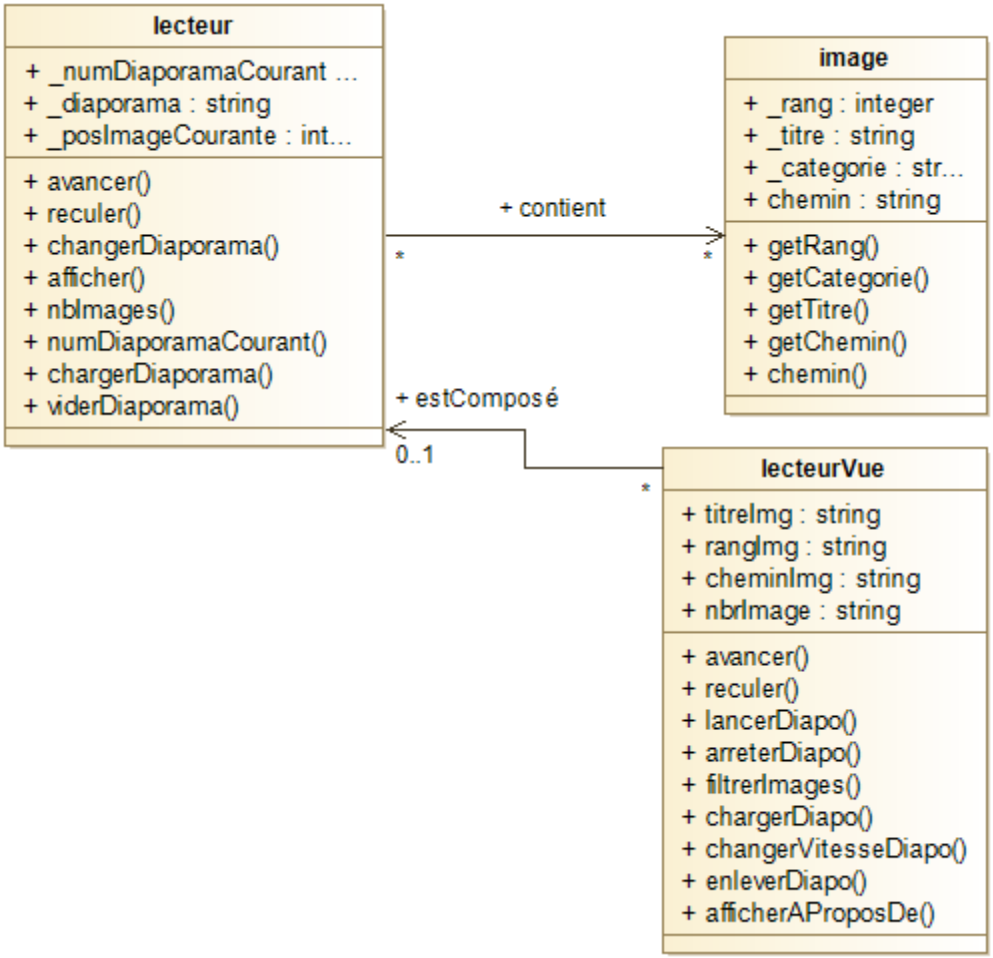


Figure 21 - Diagramme de classes (UML) - v-2

Classe LecteurVue			
Nom Attribut	**Signification**	**Type**	**Exemple**
titreImg	Le titre de l'image actuelle, affichée à l'écran	String	'Mickey'
rangImg	Le rang de l'image affichée dans son diaporama	String	'2'
cheminImg	Le chemin d'accès à l'image actuelle	String	':/lecteurDiapo/cartesDisney/Disney_30.gif'
nbrImage	Le nombre total d'images présentes dans le diaporama	String	'4'

Tableau 3 : Dictionnaire des éléments - Classe LecteurVue

```
#ifndef LECTEURVUE_H
#define LECTEURVUE_H

#include <QMainWindow>
#include "lecteur.h"


QT_BEGIN_NAMESPACE
namespace Ui { class lecteurvue; }
QT_END_NAMESPACE

class lecteurvue : public QMainWindow

{
    Q_OBJECT

public:
    lecteurvue(QWidget *parent = nullptr);
    ~lecteurvue();

private:
    Ui::lecteurvue *ui;
    Lecteur* monLecteur = new Lecteur();

private :
    QString titreImg;
    QString rangImg;
    QString cheminImg;
    QString nbrImage;

public slots :
```

```
    void avancer();
    void reculer();
    void majImage();
    void lancerDiapo();
    void arreterDiapo();
    void filtrerImages();

    void chargerDiapo();
    void changerVitesseDiapo();
    void enleverDiapo();
    void afficherAProposDe();
};


#endif // LECTEURVUE_H
```

Figure 22 - Schéma de Classe - LecteurVue

Comportement de l'application

7.1 Diagramme états-transitions-actions du lecteur de diaporamas (v2)

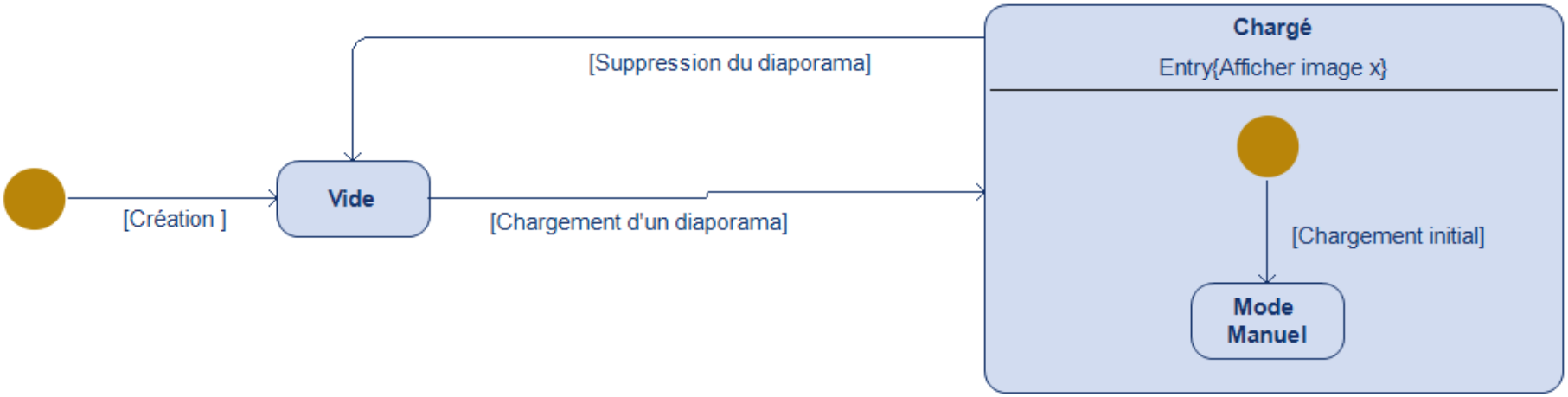


Figure 23 : Diagramme états-transitions du lecteur de diaporamas – v2

On considère que tous les états sont de potentiels états de fin, puisque l'utilisateur peut fermer l'application dès qu'il le souhaite. C'est pour cette raison qu'aucun symbole n'a été ajouté au diagramme d'état pour signaler la fin du programme.

7.2 Dictionnaire des états, événements et Actions (v2)

Dictionnaire des états du diaporama

nomEtat	*Signification*
`Vide`	Le lecteur est créé mais ne contient pas encore de diaporama.
`Chargé`	Le lecteur contient un diaporama.
`Mode Manuel`	Le lecteur est chargé d'un diaporama et est en mode manuel, c'est-à-dire que le déplacement entre les images se fait grâce aux boutons.

Tableau 4 : États du lecteur de diaporamas – v2

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`Création`	Création du lecteur lors du lancement du programme
`Chargement d'un diaporama`	Chargement d'un diaporama via l'application
`Chargement initial`	Lors du chargement du diaporama, l'état du diaporama est automatiquement en mode manuel.
`Supression du diaporama`	L'utilisateur retire le diaporama du lecteur.

Tableau 5 : Evénements faisant changer le diaporama d'état – v2

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*
`Afficher image x`	Lorsque le diaporama est chargé dans le lecteur, la première image ayant pour titre x est affiché à l'écran avec les informations nécessaires et correspondantes.

Tableau 6 : Actions à réaliser lors des changements d'état – lecteur de diaporamas v2

7.3 Table T_EtatsEvenementsActions (v2)

Correspondance matricielle du diagramme états-transitions de l'application :
- en *ligne : les **états*** du lecteur de diaporamas (éventuel état de départ d'une transition)
- en *colonne : les **événements*** faisant changer le lecteur d'état (déclencheur d'une transition)
- dans chaque cellule : l'état d'arrivée de la transition + action/traitement à faire + éventuellement garde accompagnant la transition

Élément graphique prenant en charge cet événement →	Pas d'éléments graphiques → Lancement du programme	actionCharger_Diaporama	Pas d'éléments graphiques mais impliqué par actionCharger_Diaporama	actionEnlever_Diaporama
nomEtat *Événement* →	Création	Chargement d'un diaporama	Chargement initial	Suppression du diaporama
Vide	X			X
Chargé		X		
Mode manuel			X Afficher image x	

Tableau 7 : Matrice d'états-transitions du lecteur de diaporamas – v2

Implémentation et tests

8.1 Implémentation (v2)

Liste et rôle des fichiers de cette version :

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas Définition de la structure et du comportement de la fenêtre du lecteur de diaporama
lecteurVue.cpp	Corps de la classe LecteurVue.
lecteurvue.ui	Fichier du dessin de l'interface réalisé par QtDesigner
lecteur.h	Spécification de la classe Lecteur. Définition des méthodes du lecteur, qui permettent de faire fonctionner comme

	souhaité le lecteur de diaporama
lecteur.cpp	Corps de la classe Lecteur
image.h	Spécification de la classe Image Définition des propriétés et des méthodes spécifiques aux images composant les diaporamas.
image.cpp	Corps de la classe Image
main.cpp	Fichier main.cpp Lance le processus

Remarques sur l'implémentation :

Les connexions ont été rajoutés pour les actions de la barre supérieure pour les chargements et retraits de diaporama, le changement de vitesse de développement, la boite de message pour les informations et la fermeture de la fenêtre.

8.2 Tests (v2)

Pour réaliser les tests, nous nous sommes préoccuper de l'enchainement nécessaire pour considérer que le programme est fonctionnel.

La 1ère image affichée doit être Pluto, suivi de Cendrillon, Blanche Neige et Mickey. On réalise ainsi des tests similaires à ceux de la v-0.

Pour tester la fonctionnalité « Quitter » et « A Propos De », nous devions simplement vérifier que le résultat était celui attendu.

Testeur :	M.Guiheneuf	**Date :**	22/05/2023	
Element Testé :	LecteurDiaporama : v-2	**Version :**	1.0	

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	Résultats Finaux	Conforme
Valide n°1	Image de départ	-	Image et informations de Pluto	Pluto	
Valide n°2	Avancer ==> 1ère fois	-	Cendrillon	Cendrillon	
Valide n°3	Avancer ==> 2ème fois	-	Blanche Neige	Blanche Neige	
Valide n°4	Avancer ==> 3ème fois	-	Mickey	Mickey	
Valide n°5	Avancer ==> 4ème fois	-	Pluto	Pluto	
Valide n°6	Reculer ==> 1ère fois	-	Mickey	Mickey	
Valide n°7	Reculer ==> 2ème fois	-	Blanche Neige	Blanche Neige	
Valide n°8	Reculer ==> 3ème fois	-	Cendrillon	Cendrillon	
Valide n°9	Reculer ==> 4ème fois	-	Pluto	Pluto	
Valide n°10	Fermeture du programme par le menu Fichier > Quitter	-	Fermeture du programme	Fermeture du programme	
Valide n°11	Ouverture d'une boite de dialogue par le menu Aide > A propos de	-	Ouverture de la boite de dialogue	Ouverture de la boite de dialogue	

Figure 24 - Test v-2

Version v3 –

Dans la version 3 du code nous avons mis en œuvre le mode automatique, le lecteur dispose donc d'un mode de lecture automatique des images grâce au bouton "Lancer Diaporama" et nous pouvons arrêter ce mode en cliquant sur le bouton "Arrêter Diaporama", "Avancer" ou "Reculer", comme précisé dans le sujet. Par ailleurs, en mode automatique, une utilisation du bouton « Lancer Diaporama » relance le diaporama à la première image. Pour mettre en œuvre ce mode automatique, un timer est utilisé, initialisé à 2 secondes (2000 millisecondes) qui démarre lorsque le bouton est enclenché. Lorsque ce timer est écoulé, on avance ce qui a pu se faire avec une connexion.

Le lancement du diaporama doit aussi déconnecter les boutons « Avancer » et « Reculer » pour les connecter avec l'arrêt du diaporama. Cependant, ils doivent être reconnecter normalement dès que le mode automatique est arrêté.

Version v4 –

La version 4 consistait à permettre à l'utilisateur de paramétrer la vitesse de défilement qu'il souhaite. Il fallait ainsi une fenêtre de dialogue, dans laquelle l'utilisateur choisit entre 1 et 10 secondes d'intervalle entre les images. La valeur saisie est récupérée dans une variable choixVitesseDef, qui va être multipliée par 1000 pour donner le temps en millisecondes et définir un nouvel intervalle pour le timer avec la variable vitesseDef.

Version v5 –

Diagramme de classes (UML)

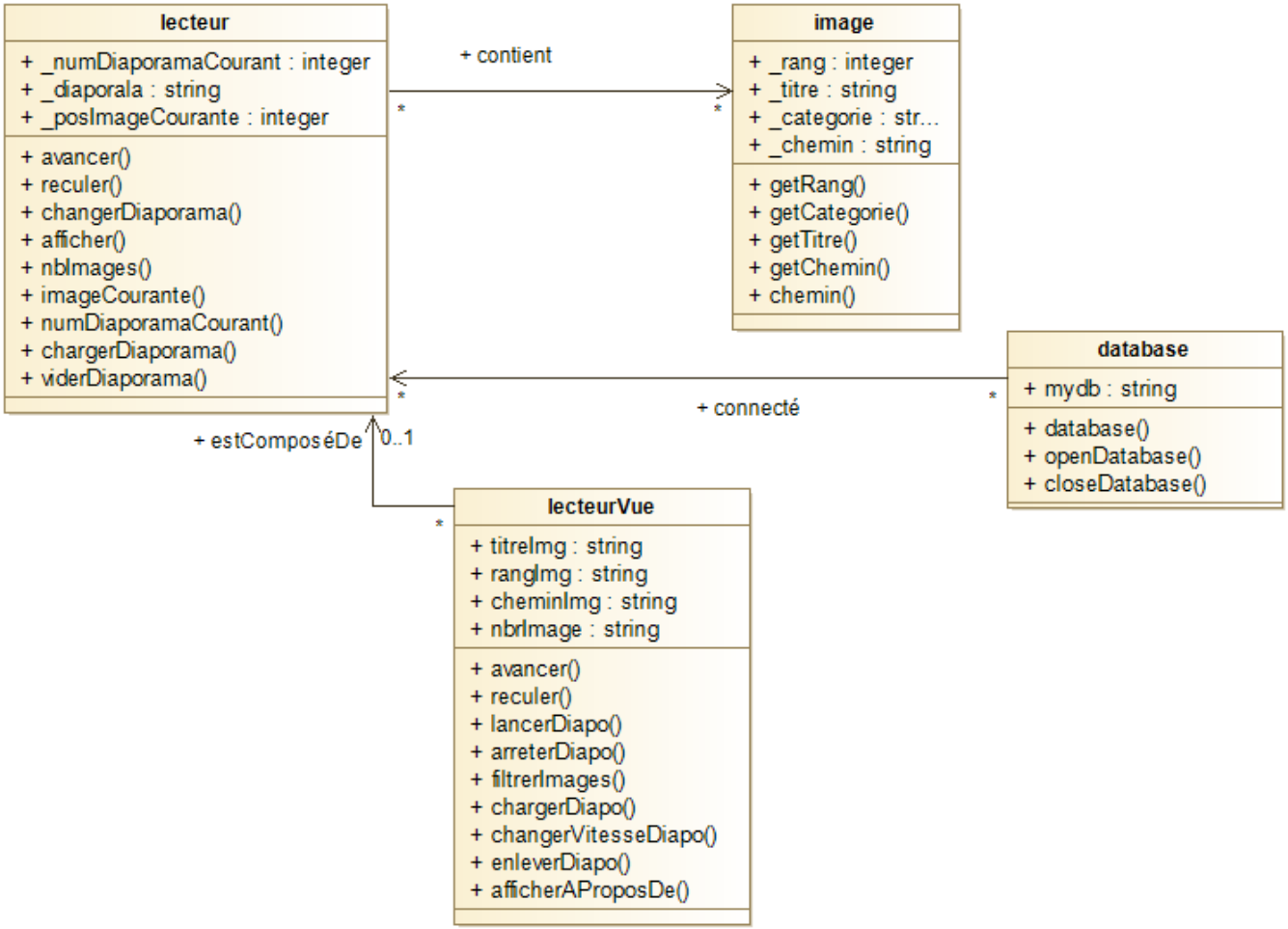


Figure 25 - Diagramme de classes (UML) - v-5

Classe database			
Nom Attribut	**Signification**	**Type**	**Exemple**
mydb	La base de données que nous allons utiliser pour obtenir nos informations sur les images à afficher	QSqlDatabase	'bd_nodenot9'

Figure 26 - Diagramme des élements - Classe database

```
#ifndef DATABASE_H
#define DATABASE_H

#include <QSqlDatabase>

#define DATABASE_NAME "BD_Diapo_Lakartxela"
#define CONNECT_TYPE "QODBC"

class database
{
public:
    database();
    bool openDatabase();
    void closeDatabase();

private :
    QSqlDatabase mydb;
};

#endif // DATABASE_H
```

Figure 27 - Schéma de Classe - Database

Comportement de l'application

11.1 Diagramme états-transitions-actions du lecteur de diaporamas (v5)

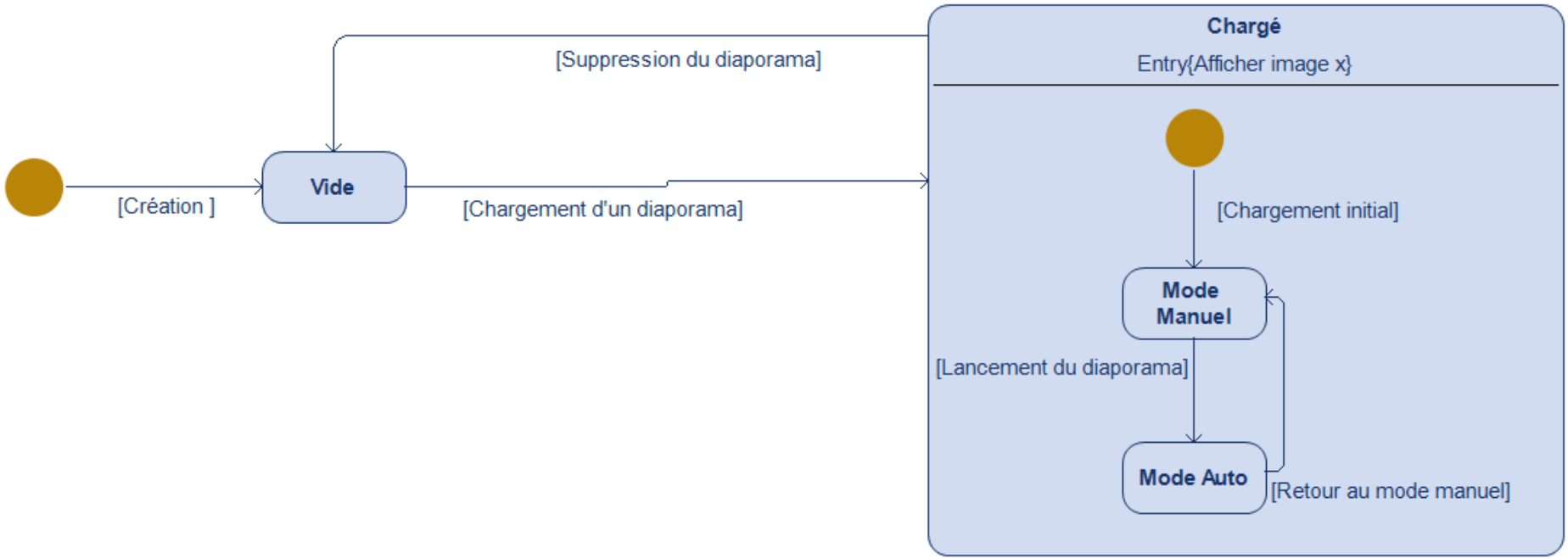


Figure 28 : Diagramme états-transitions du lecteur de diaporamas – v5

11.2 Dictionnaire des états, événements et Actions (v5)

Dictionnaire des états du diaporama

nomEtat	*Signification*
`Vide`	Le lecteur est créé mais ne contient pas encore de diaporama.
`Chargé`	Le lecteur contient un diaporama.
`Mode Manuel`	Le lecteur est chargé d'un diaporama et est en mode manuel, c'est-à-dire que le déplacement entre les images se fait grâce aux boutons.
`Mode Auto`	Le lecteur est chargé d'un diaporama et est en mode Auto, c'est-à-dire que le déplacement entre les images se fait automatiquement grâce à une minuterie.

Tableau 8 : États du lecteur de diaporamas – v5

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`Création`	Création du lecteur lors du lancement du programme
`Chargement d'un diaporama`	Chargement d'un diaporama via l'application
`Chargement initial`	Lors du chargement du diaporama, l'état du diaporama est automatiquement en mode manuel.
`Supression du diaporama`	L'utilisateur retire le diaporama du lecteur.
`Lancement du diaporama`	L'utilisateur passe au mode automatique en cliquant sur le bouton dédié au lancement.
`Retour au mode manuel`	Une fois en mode auto, l'utilisateur peut repasser en mode manuel comme initialement.

Tableau 9 : Evénements faisant changer le diaporama d'état – v5

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*
`Afficher image x`	Lorsque le diaporama est chargé dans le lecteur, la première image ayant pour titre x est affiché à l'écran avec les informations nécessaires et correspondantes.

Tableau 10 : Actions à réaliser lors des changements d'état – lecteur de diaporamas v5

11.3 Table T_EtatsEvenementsActions (v5)

Correspondance matricielle du diagramme états-transitions de l'application :
- en *ligne : les **états*** du lecteur de diaporamas (éventuel état de départ d'une transition)
- en *colonne : les **événements*** faisant changer le lecteur d'état (déclencheur d'une transition)
- dans chaque cellule : l'état d'arrivée de la transition + action/traitement à faire + éventuellement garde accompagnant la transition

Élément graphique prenant en charge cet événement →	Pas d'éléments graphiques → Lancement du programme	actionCharger_Diaporama	Pas d'éléments graphiques mais impliqué par actionCharger_Diaporama	actionEnlever_Diaporama	bLancerDiapo	bArreterDiapo, bAvancer, bReculer
Événement → *nomEtat*	Création	Chargement d'un diaporama	Chargement initial	Suppression du diaporama	Lancement du diaporama	Retour au mode manuel
Vide	X			X		
Chargé		X				
Mode manuel			X Afficher image x			X
Mode Auto					X	

Tableau 11 : Matrice d'états-transitions du lecteur de diaporamas – v5

Implémentation et tests

12.1 Implémentation (v5)

Liste et rôle des fichiers de cette version :

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas Définition de la structure et du comportement de la fenêtre du lecteur de diaporama
lecteurVue.cpp	Corps de la classe LecteurVue.

lecteurvue.ui	Fichier du dessin de l'interface réalisé par QtDesigner
lecteur.h	Spécification de la classe Lecteur. Définition des méthodes du lecteur, qui permettent de faire fonctionner comme souhaité le lecteur de diaporama
lecteur.cpp	Corps de la classe Lecteur
image.h	Spécification de la classe Image Définition des propriétés et des méthodes spécifiques aux images composant les diaporamas.
image.cpp	Corps de la classe Image
main.cpp	Fichier main.cpp Lance le processus
Database.h	Spécification de la classe Database Définition des méthodes d'accès à la base de données
Database.cpp	Corps de la classe Database

Remarques sur l'implémentation :

Initialement, les images étaient dans le fichier lecteur.cpp en dur pour les afficher et tester les précédentes fonctionnalités. On fait appel maintenant à la base de données grâce à une requête nous renvoyant l'idPhoto, le nom de la catégorie, le titre de l'image et le chemin d'accès à l'image. On récupère ces informations dans le but de créer une image à charger à partir de la classe Image. Celle-ci sera poussé dans le diaporama que l'on devra afficher.

Nous avons légèrement modifié le chemin récupéré afin d'afficher nos images à partir des ressources du fichier.

12.2 Tests (v5)

Pour tester le fonctionnement de cette nouvelle version, nous devons afficher les images issus de la base de données, en restant sûr du fait que les fonctionnalités précédentes fonctionnent toujours.

Testeur :	M.Guiheneuf	**Date :**	03/05/2023
Element Testé :	LecteurDiaporama : v-5	**Version :**	1.0

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	Résultats Finaux	Conforme
Valide n°1	Image de départ	-	Disney_0.gif	Disney_0.gif	
Valide n°2	Avancer ==> 1 fois	-	Disney_1.gif	Disney_1.gif	
Valide n°3	Reculer ==> 1 fois	-	Disney_0.gif	Disney_0.gif	
Valide n°4	Lancement en mode Auto	-	Fonctionnement normal	Fonctionnement normal	
Valide n°5	Arrêt avec bArreterDiapo	-	Fonctionnement normal	Fonctionnement normal	
Valide n°6	Arrêt avec bReculer	-	Fonctionnement normal	Fonctionnement normal	
Valide n°7	Arrêt avec bAvancer	-	Fonctionnement normal	Fonctionnement normal	

Figure 29 - Test v-5

Version v6 –

Bilan

Dépôt Git où trouver le projet complet (les versions réalisées)
https://github.com/mattin-guiheneuf/LecteurDiaporama
Temps global de travail (pour le groupe)
Apprentissages majeurs
Difficultés majeures
Points positifs / négatifs de l'activité