CONGUISTI Nicolas -TD1/TP1
DUVIGNAU Yannis -TD1/TP1
VICTORAS Dylan -TD1/TP1

Date de rendu : 05/06/2023

S2.01
Développement d'une application

Dossier d'analyse et conception

Dossier réalisé de la V0 à la V6, la V7 en cours d'implémentation

Table des matières

1. Compléments de spécifications externes.

Lors de la première séance, nous avons conceptualisé l'interface de l'application. Nous avions pour ressource un document qui décrivait les boutons, labels et fonctionnalités présentes dans l'application.

Ce document nous a paru assez flou, certaines fonctionnalités n'étaient pas mises en valeur, elles étaient cachées ou même absentes du document.

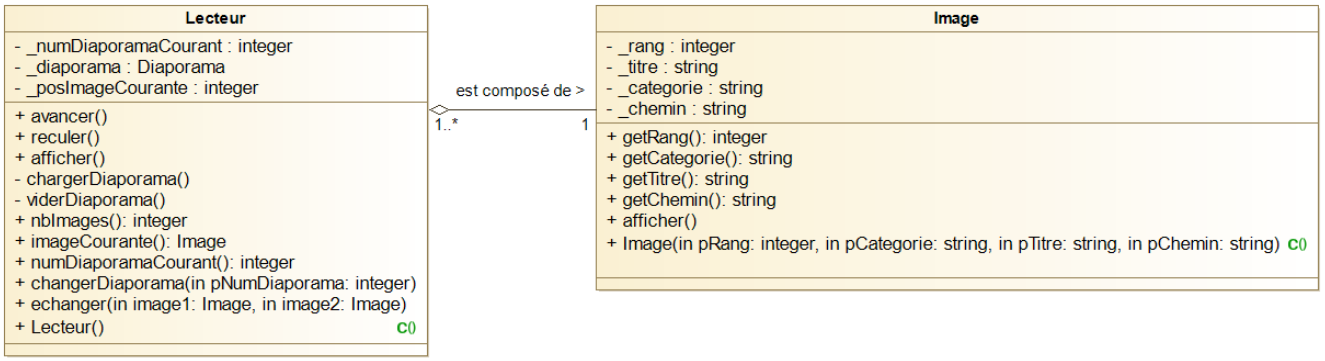
Ces difficultés nous ont ralenti dans l'avancement du projet. A part ça, nous n'avons rien à signaler concernant cette SAE.

2. Scénarios

Cas d'utilisation	**Lire un diaporama**		
Acteur primaire	Utilisateur		
Système	Lecteur de diaporama		
Intervenants			
Niveau	Objectif utilisateur		
Préconditions	L'utilisateur a ouvert l'application		
Opérations	**Utilisateur**	**Lecteur de diaporama**	***Interfaces***
1		Le système affiche le lecteur vide, aucun diaporama n'est chargé.	***1. Fenêtre principale***
2	L'utilisateur charge un diaporama stocké dans une base de données.		
3		Le système passe en mode manuel	
4		Le système affiche la première image avec sa catégorie, son titre et son rang dans le diaporama.	
5	L'utilisateur défile les images du diaporama (les images suivantes ou précédentes de l'image courante).		
6		Le système affiche les images souhaitées.	
7	L'utilisateur choisit de quitter l'application.		
Extensions			
5.A	**L'utilisateur choisit d'activer le mode automatique.**		
5.A.1	L'utilisateur choisit d'activer le mode automatique.		***Fenêtre principale***
5.A.2		Le système affiche les images de manière automatique, sans action de l'utilisateur, avec une vitesse de défilement prédéfinie.	
5.A.3	*Retour à l'étape 7		
5.B	**L'utilisateur choisit de changer la vitesse de défilement.**		
5.B.1	L'utilisateur choisit de changer la vitesse de défilement.		***Fenêtre principale***
5.B.2		Le système affiche une fenetre de dialogue avec plusieurs vitesses de défilement prédéfinies.	***2. Fenêtre de paramétrage de la vitesse de défilement***
5.B.3	L'utilisateur choisi une vitesse de défilement, puis confirme.		
5.B.4		Le système enregistre la vitesse de défilement souhaitée.	
5.B.5		Le système affiche les images de manière automatique, sans action de l'utilisateur, avec la vitesse de défilement définie par l'utilisateur.	***Fenêtre principale***
5.A.6	*Retour à l'étape 7		

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.



(b) Dictionnaire des éléments pour chaque classe

Classe Image			
Nom attribut	**Signification**	**Type**	**Exemple**
`_rang`		`unsigned int`	`3`
`_titre`		`string`	`"Blanche Neige"`
`_categorie`		`string`	`"personne"`
`_chemin`		`string`	`"C:\\cartesDisney\\carteDisney2.gif"`

Classe Lecteur			
Nom attribut	**Signification**	**Type**	**Exemple**
`_numDiaporamaCourant`		`unsigned int`	`0`

_diaporama		Diaporama	1
_posImageCourante		unsigned int	3

(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 4 : Schéma de classes = Classe XXX

(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

4. Description de la version

Affichage du diaporama en brut sur le terminal. Seulement les méthodes avancer() et reculer() sont présentes.

5. Implémentation et tests

5.1 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

5.2 Test

Test du bon affichage du diaporama en brut dans le terminal

Testeur	Nicolas Conguisti		**Date**	19/05/23
Élement testé	Lecteur de diaporama		**Version**	0
Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	
Valide n°1	Le diaporama complet s'affiche correctement en brut dans le terminal.	Aucune	[Figure 1] et [Figure 2]	
Invalide n°1	Avancer() ne s'affiche pas correctement en brut dans le terminal.	Aucune	Échec	
Invalide n°2	Reculer() ne s'affiche pas correctement en brut dans le terminal.	Aucune	Échec	

Figure 1 : Avancer ()

Résultats attendus :

```
Test avancer() : 4 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:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
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:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
```

Résultats obtenus :

```
Test avancer() : 4 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:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
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:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
```

Figure 2 : Reculer ()

Résultats attendus :

```
Test reculer() : 5 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:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
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:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
reculer() :
Diaporama num. 1
image courante : image( rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)
```

Résultats obtenus :

```
Test reculer() : 5 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:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
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:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
reculer() :
Diaporama num. 1
Image courante : image( rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)
```

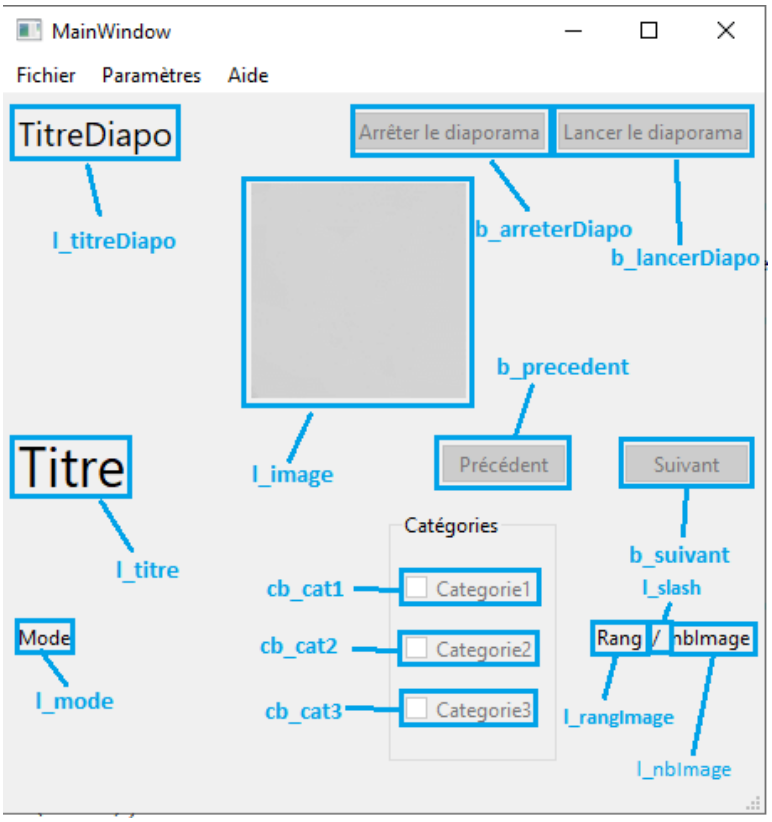
Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

Version v1

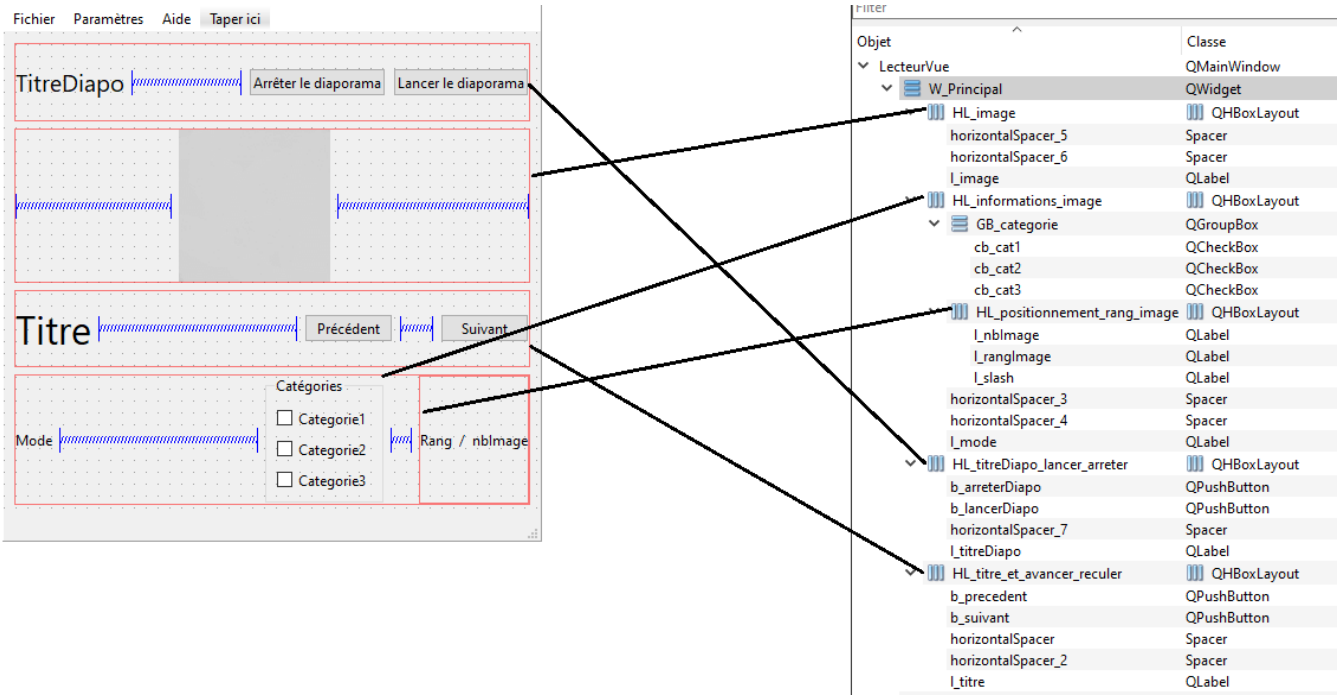
6. Description de la version

Création de l'interface graphique non fonctionnelle. L'appel des boutons provoque des signaux sur le terminal.

7. Éléments d'interface



Objets Graphiques



Les layouts

8. Implémentation et tests

8.1 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 :

Commenter brièvement les choix importants d'implémentation réalisés, comme par exemple, les signals/slots

Tous les éléments de l'interface sont connectés de la manière suivante :

```
QObject::connect(ui->suivant,SIGNAL(clicked()),this,SLOT(avancer()));
```

Les éléments du menu, quant à eux, sont connectés comme ceci :

```
QObject::connect(ui->chargerDiapo,SIGNAL(triggered()),this,SLOT(chargerDiapo()));
```

Il est important de noter que les boutons cb_cat1, cb_cat2 et cb_cat3 ne sont pas fonctionnels. En effet, les cases à cocher ne devraient produire une action que lorsqu'elles sont cochées, Or, dans le cas présent, elles produisent une action dans tous les cas, même quand elles sont décochées.

Nous corrigerons ce problème dans une version ultérieure.

8.2 Test

8.2.1 Test sur l'affichage de tous les éléments de la fenêtre

Testeur	Nicolas Conguisti		**Date**	19/05/23
Élement testé	Lecteur de diaporama		**Version**	1

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)
Valide n°1	Le diaporama s'affiche correctement en version graphique.	Aucune	Affichage complet et lisible de tous les éléments du diaporama.
Invalide n°1	Le diaporama ne s'affiche pas correctement en version graphique.	Aucune	Échec ou affichage incomplet.

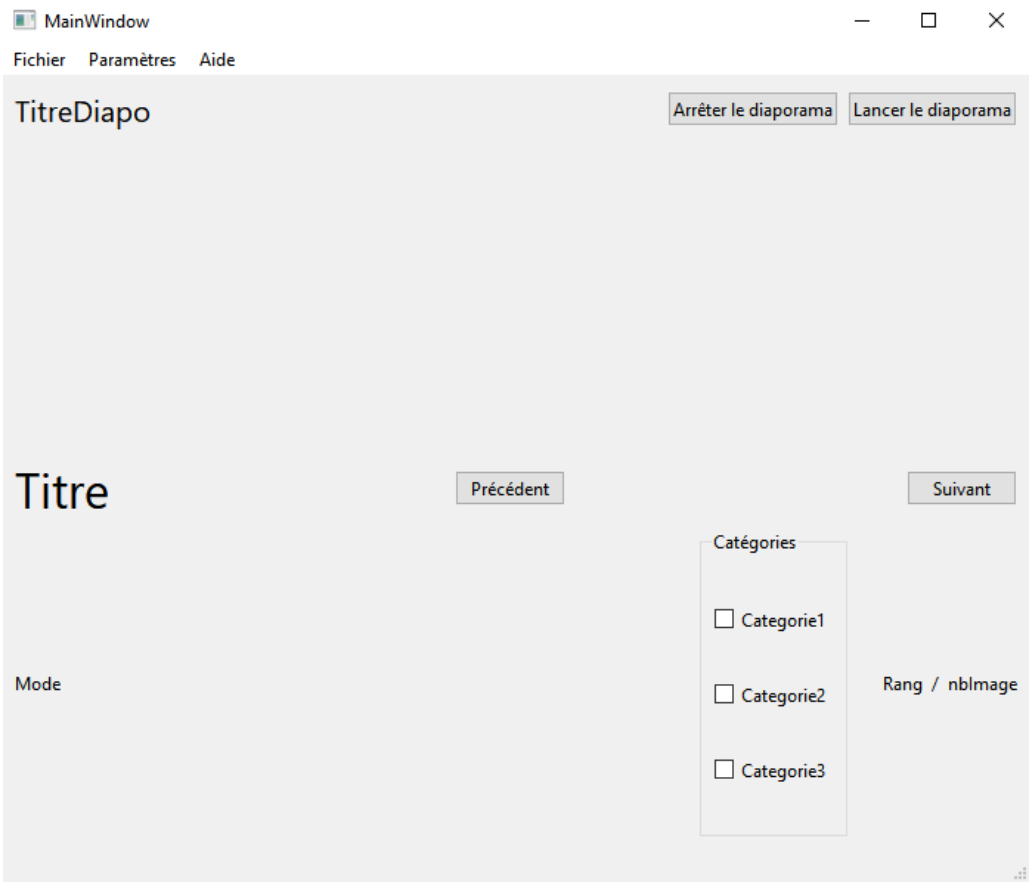
Cette version du lecteur n'affiche pour l'instant pas les images. Cela sera corrigé dans les versions suivantes. Tous les autres éléments s'affichent correctement, le test est donc validé.

8.2.2 Test sur l'étirement de la fenêtre

Ce test montre la capacité de la fenêtre à être responsive. Elle doit s'adapter à la taille que l'utilisateur lui donne.

Testeur	Nicolas Conguisti		**Date**	19/05/23
Élement testé	Lecteur de diaporama		**Version**	1

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)
Valide n°1	Test sur le bon étirement de la fenêtre	Aucune	Tous les éléments de la fenêtre s'affichent de manière "responsive".
Invalide n°1		Aucune	Les éléments de la fenêtre ne s'affichent pas tous de manière "responsive".



La fenêtre s'affiche de manière responsive lorsqu'on l'agrandit ou la rétrécie. Le test est donc validé.

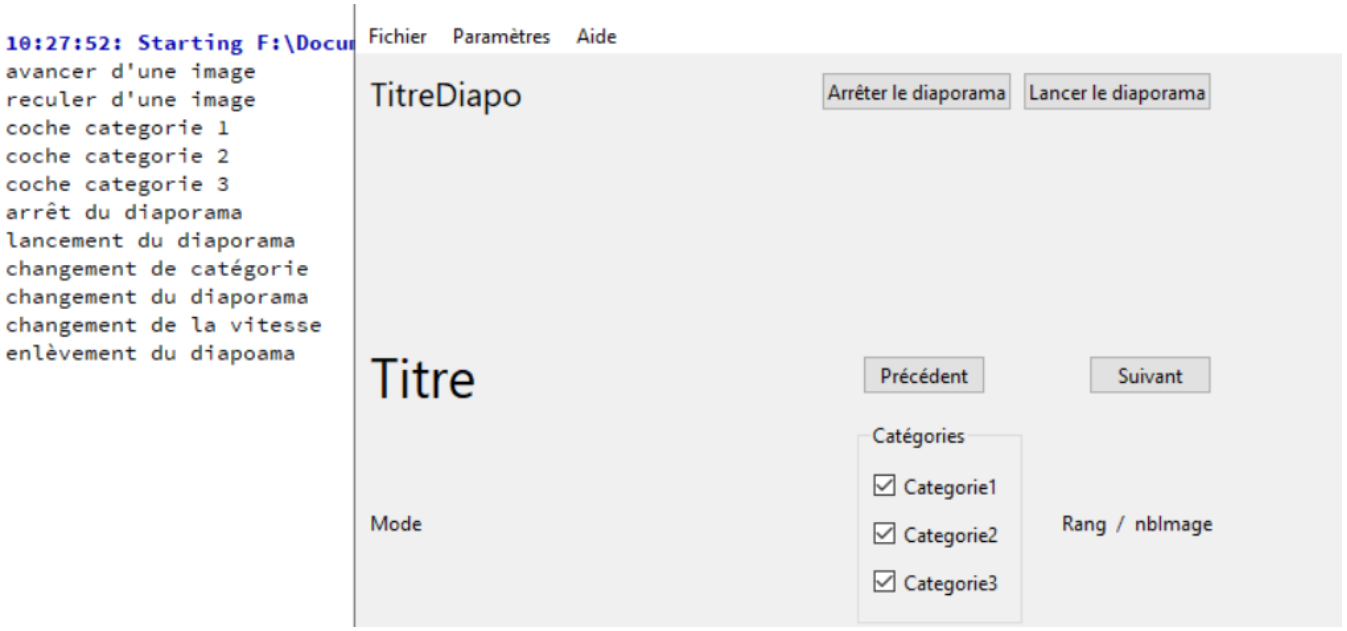
8.2.3 Test sur le signal produit par les boutons

Ce test vise à savoir si tous les boutons produisent un signal quand on les pressent.

Testeur	Nicolas Conguisti		**Date**	19/05/23
Élement testé	Lecteur de diaporama		**Version**	1

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)
Valide n°1	Le diaporama s'execute correctement en version graphique.	Aucune	Tous les éléments du diaporama produisent un signal à leur appel.
Invalide n°1	Le diaporama ne s'execute pas correctement en version graphique.	Aucune	Tous les éléments du diaporama ne produisent pas un signal à leur appel.

Pour ce test, nous avons fait le choix d'activer tous les boutons du diaporama, puis de vérifier qu'un affichage se soit produit sur la console. Nous avons obtenu les résultats suivants.

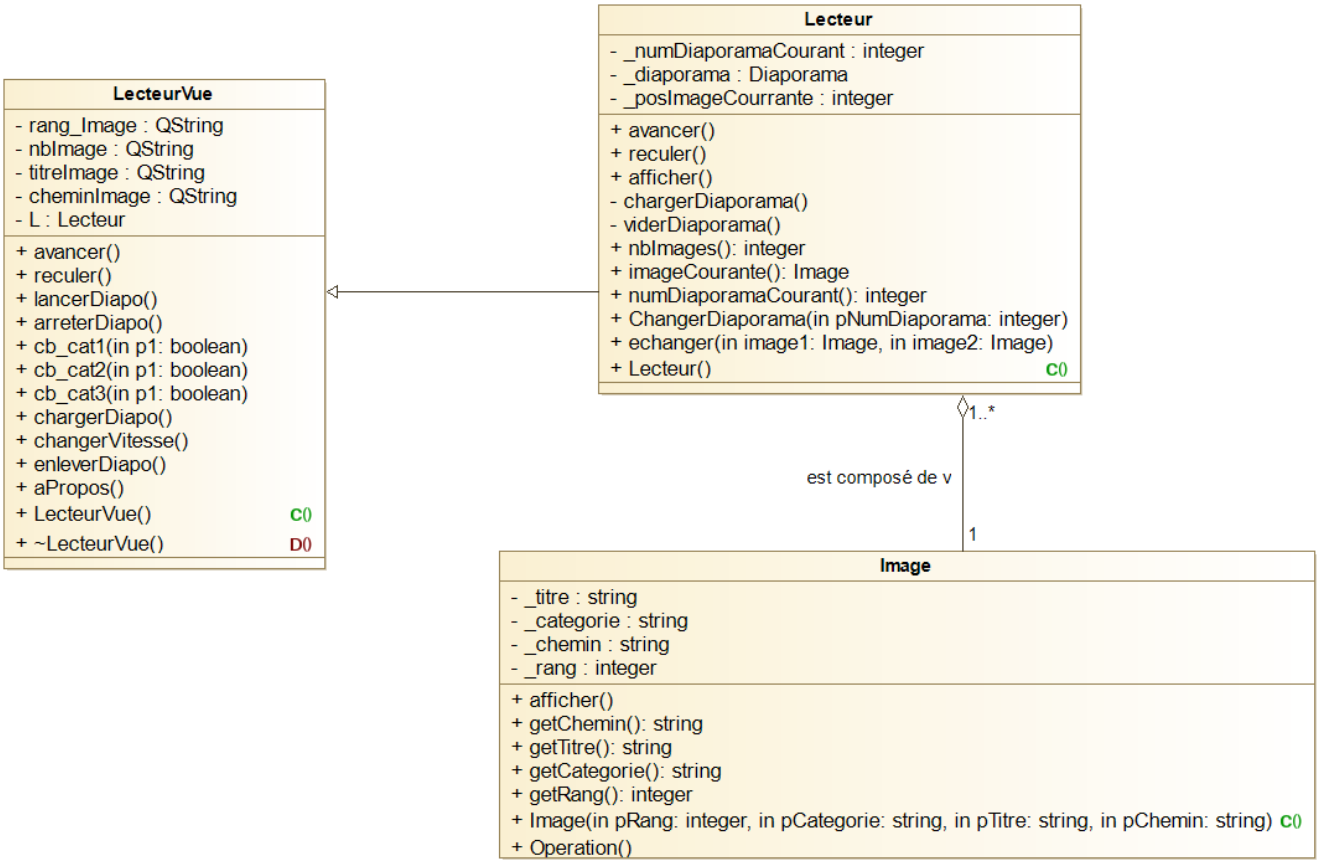


Tous les éléments du diaporama produisent un signal à leur appel. Le test est donc validé.

Version v2

Cette version est une liaison de la version 0 et 1, elle permet d'obtenir une première version complètement fonctionnelle.

9. Diagramme de classes (UML)



Commentaire :

Ajout d'une classe lecteurVue qui hérite des méthodes de la classe Lecteur

(b) Dictionnaire des nouveaux éléments apportés pour chaque classe

Nom attribut	Signification	Type	Exemple	classe
`L`		`Lecteur`		`LecteurVue`
`rangImage`		`QString`	`1`	`LecteurVue`

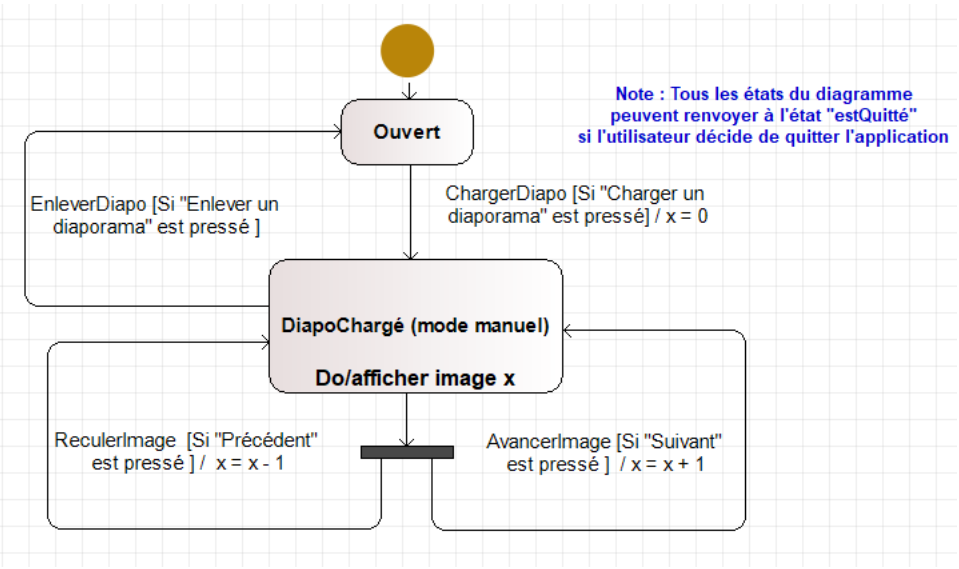
nbImage		QString	10	LecteurVue
titreImage		QString	Disney_0.gif	LecteurVue
cheminImage		QString	"C:\\cartesDisney\\carteDisney2.gif"	LecteurVue

10. Comportement de l'application

Amélioration apportées à la version 1 :

- *Liaison de la V0 et de la V1.*
- *Création d'un spacer entre le bouton précédent et le bouton suivant pour améliorer la responsivité de la fenêtre.*
- *Les cases à cocher ne déclenchent un signal que lorsqu'elles sont cochées (dans les versions ultérieures, un signal était aussi déclenché quand les cases étaient décochées).*

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



10.2 *Dictionnaire des états, événements et Actions (v2)*

Dictionnaire des états du diaporama

NomEtat	*Signification*
`Ouvert`	*Il est déclenché quand l'utilisateur ouvre le lecteur de diaporama. A ce stade, aucun diaporama n'est chargé.*
`DiapoChargé(mode manuel)`	*Quand l'utilisateur charge un diaporama, cet état devient actif. Il entraîne l'action "afficher l'image x".*

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`EnleverDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"quand l'utilisateur décide d'enlever le diaporama. Il renvoie à l'état "Ouvert".*
`ChargerDiapo`	*Cet événement est déclenché depuis l'état "Ouvert" quand l'utilisateur souhaite ouvrir un diaporama. Il renvoie à l'état "DiapoChargé".*
`ReculerImage`	*Cet événement est déclenché quand l'utilisateur appuie sur le bouton "Précédent". L'image courante du diaporama est alors changée (x = x - 1), puis affichée.*
`AvancerImage`	*Cet événement est déclenché quand l'utilisateur appuie sur le bouton "Suivant". L'image courante du diaporama est alors changée (x = x + 1), puis affichée.*

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*
`Afficher image x`	*Cette action est effectuée dans l'état "DiapoChargé". A chaque fois qu'une nouvelle image est chargée, elle est donc affichée.*

10.3 *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 à	`bouton ChargerDiapo: a_chargerDiapo`	`bouton enleverDiapo: a_enleverDiapo`	`bouton quitter :a_quitter`	`bouton précédent : b_precedent`	`bouton suivant : b_suivant`
Événement à nomEtat	`ChargerDiapo`	`EnleverDiapo`	`Quitter est pressé`	`ReculerImage`	`AvancerImage`
Ouvert	`DiapoChargé, Chargement d'un diaporama`				
`DiapoChargé`		`Ouvert, Enlèvement d'un diaporama`	`EstQuitté`	`DiapoChargé, affichage d'une nouvelle image`	`DiapoChargé, affichage d'une nouvelle image`

11. Implémentation et tests

11.1 *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 contient les méthodes permettant les interactions possibles sur les objets graphiques de l'interface du lecteur du 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. contient les méthodes permettant de réaliser les actions sur le lecteur de diaporama (avancer, reculer, …).
lecteur.cpp	Corps de la classe Lecteur.
image.h	Spécification de la classe Image contient les méthodes permettant de récupérer les informations d'une image (catégorie, rang, …).
image.cpp	Corps de la classe Image
main.cpp	Permet de lancer le projet.

Remarques sur l'implémentation :

Idem V1

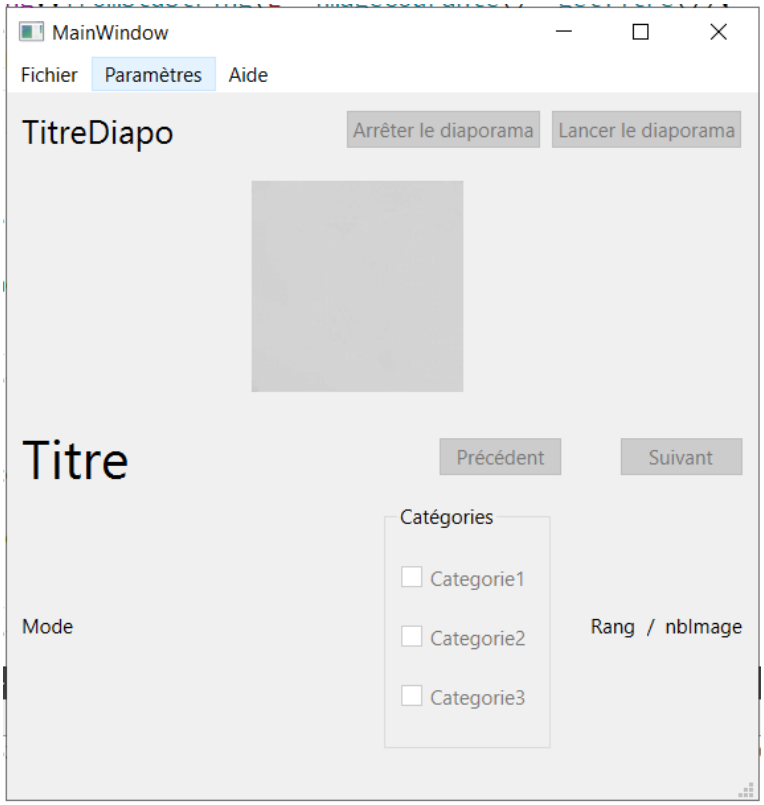
11.2 *Tests (v2)*

- Le comportement de l'interface non lié aux aspects fonctionnels du programme

11.2.1 Test sur l'affichage de l'interface

IDEM V1 : Le comportement de l'interface est identique à la version 1. Néanmoins, il est important de noter que le diaporama affiche désormais une image quand celui-ci est chargé.

Nous avons fait le choix de désactiver les boutons "suivant", "précédent", "lancer le diaporama" et "arrêter le diaporama" car tant qu'aucun diaporama n'a été chargé, il est inutile d'activer ces boutons. Aussi, une image par défaut est chargée tant qu'aucun diaporama n'a été chargé.



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

11.2.2 Test du comportement fonctionnel de l'application après l'appel des méthodes avancer() et Reculer()

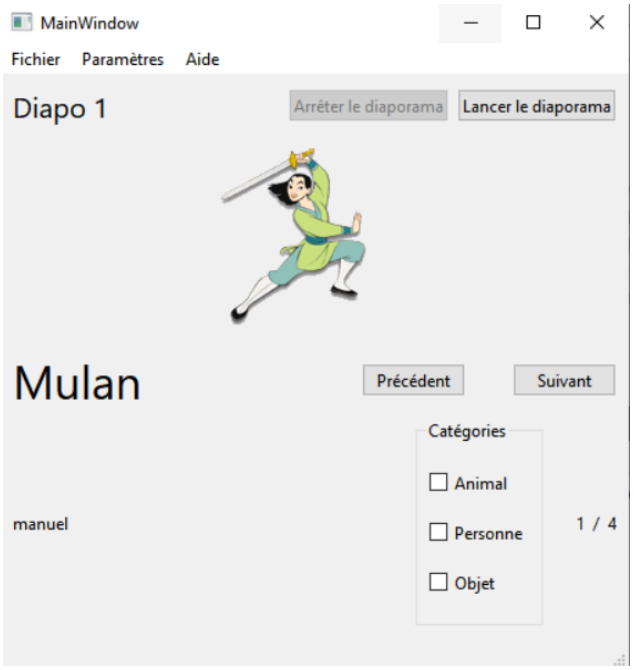
Ce test vise à déterminer si tous les éléments du diaporama sont fonctionnels et réagissent à l'appel des méthodes avancer() et Reculer()

Testeur	Nicolas Conguisti		**Date**	04/06/23
Élement testé	Lecteur de diaporama		**Version**	2

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)
Valide n°1	Test du comportement fonctionnel de l'application après l'appel des méthodes avancer() et Reculer()	Aucune	Affichage de l'mage suivante ou précédente à l'appel des méthodes avancer ou reculer.
Invalide n°1		Aucune	Échec ou affichage incomplet suite à l'appel des méthodes avancer ou reculer.

Résultats obtenus :

1ère image du diaporama



Le bouton reculer est pressé :

Le bouton avancer est pressé :

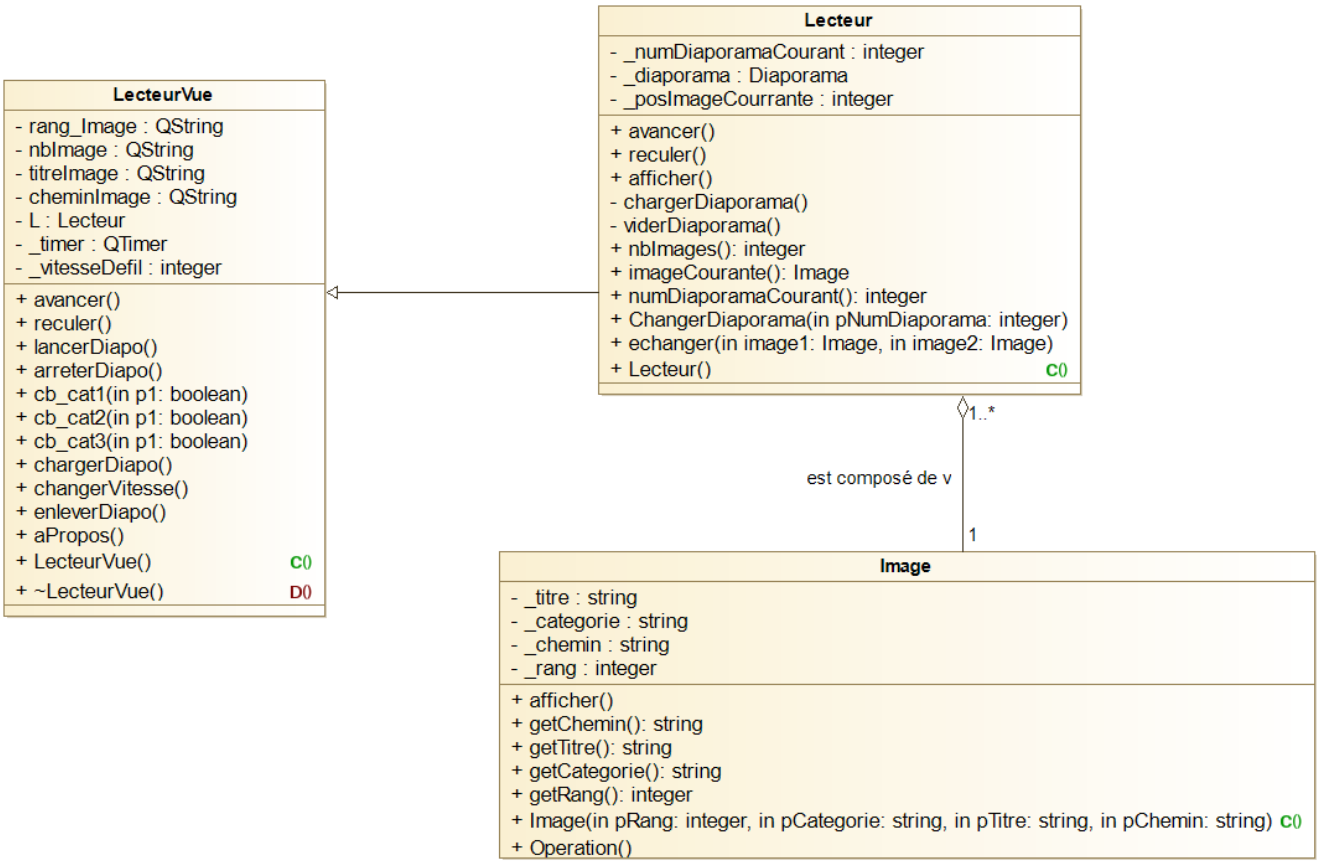
Les boutons avancer et reculer affichent de manière cohérente les images situées avant et après l'image courante. Le test est validé.

Version v3 –

12. Description de la version

Ajout du mode automatique qui prend en compte un QTimer qui déclenche la méthode avancer().
Ce timer sera déclenché dès l'activation du bouton "Lancer Diaporama".

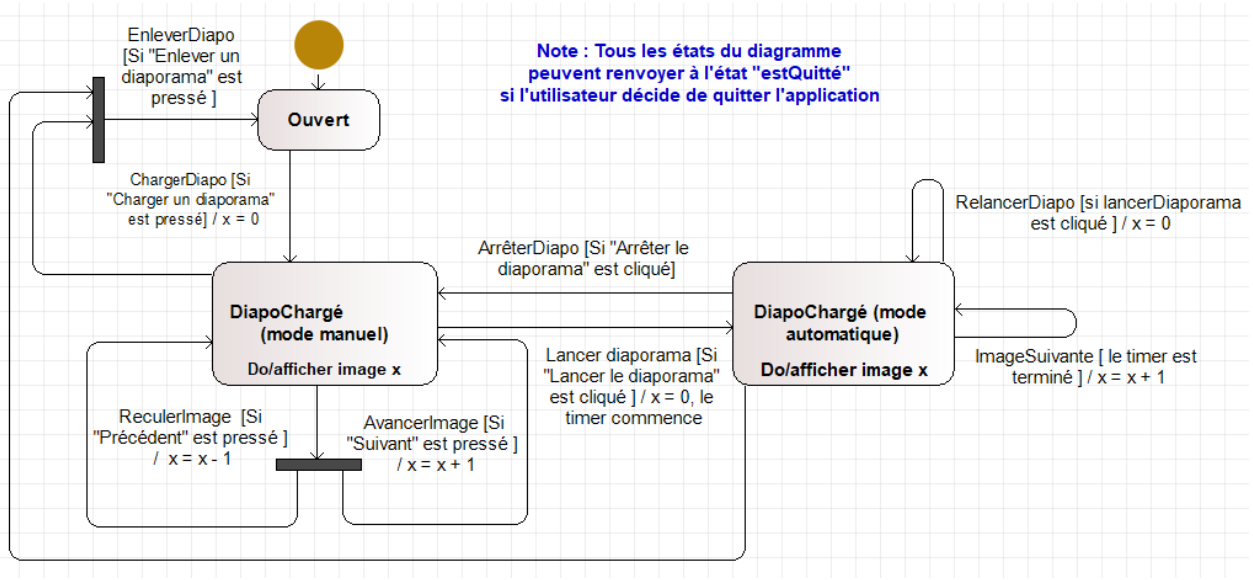
13. Diagramme de classe UML



(b) Dictionnaire des nouveaux éléments apportés pour chaque classe

Nom attribut	Signification	Type	Exemple	classe
_timer		QTimer	5000	LecteurVue
_vitesseDefil		Unsigned int	5000	LecteurVue

14. Diagramme d'état transitions



14.1 *Dictionnaire des états, événements et Actions (v3)*

Dictionnaire des états du diaporama

NomEtat	*Signification*
`Ouvert`	*Il est déclenché quand l'utilisateur ouvre le lecteur de diaporama. A ce stade, aucun diaporama n'est chargé.*
`DiapoChargé(mode manuel)`	*Un diaporama est chargé et l'utilisateur utilise le mode manuel (défilement des images à la main) de l'application.*
`DiapoChargé(mode automatique)`	*Un diaporama est chargé et l'utilisateur utilise le mode automatique (défilement automatique des images) de l'application.*

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`EnleverDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"quand l'utilisateur décide d'enlever le diaporama. Il renvoie à l'état "Ouvert".*
`ChargerDiapo`	*Cet événement est déclenché depuis l'état "Ouvert" quand l'utilisateur souhaite ouvrir un diaporama. Il renvoie à l'état*

	“DiapoChargé”.
`LancerDiapo`	*Cet événement est déclenché depuis l’état “DiapoChargé”(mode manuel) quand l’utilisateur souhaite faire défiler les images automatiquement. Il renvoie à l’état “DiapoChargé”(mode automatique).*
`ArreterDiapo`	*Cet événement est déclenché depuis l’état “DiapoChargé”(mode automatique) quand l’utilisateur souhaite arrêter de faire défiler les images automatiquement. Il renvoie à l’état “DiapoChargé”(mode manuel).*
`RelancerDiapo`	*Cet événement est déclenché depuis l’état “DiapoChargé”(mode automatique) quand l’utilisateur souhaite relancer le diaporama. Il renvoie à l’état “DiapoChargé”(mode automatique).*
`ImageSuivante`	*Cet événement est déclenché depuis l’état “DiapoChargé”(mode automatique) quand le timer arrive à 0(timer terminé). Il renvoie à l’état “DiapoChargé”(mode automatique).*
`ReculerImage`	*Cet événement est déclenché quand l’utilisateur appuie sur le bouton “Précédent”. L’image courante du diaporama est alors changée, puis affichée.*
`AvancerImage`	*Cet événement est déclenché quand l’utilisateur appuie sur le bouton “Suivant”. L’image courante du diaporama est alors changée, puis affichée.*

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*
`Afficher image x`	*Cette action est effectuée dans l’état “DiapoChargé”. A chaque fois qu’une nouvelle image est chargée, elle est donc affichée.*

14.2 *Table T_EtatsEvenementsActions (v3)*

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 à	bouton enleverDiapo: a_enleverDiapo	bouton ChargerDiapo: a_chargerDiapo	bouton lancerDiapo a_lancerDiapo	bouton arreterDiapo: b_arreterDiapo	bouton lancerDiapo a_lancerDiapo		bouton précédent : b_precedent	bouton suivant : b_suivant
Événement à *nomEtat*	EnleverDiapo	ChargerDiapo	LancerDiapo	ArreterDiapo	RelancerDiapo	ImageSuivante	ReculerImage	AvancerImage
Ouvert		DiapoChargé (mode manuel), Chargement d'un diaporama						
DiapoChargé (mode manuel)	Ouvert, Enlèvement d'un diaporama		DiapoChargé, affichage de la première image du diaporama	DiapoChargé, affichage d'une nouvelle image			DiapoChargé, affichage d'une nouvelle image	DiapoChargé, affichage d'une nouvelle image
DiapoChargé (mode automatique)					DiapoChargé, affichage de la première image du diaporama	DiapoChargé, affichage d'une nouvelle image		

15. Implémentation et Tests

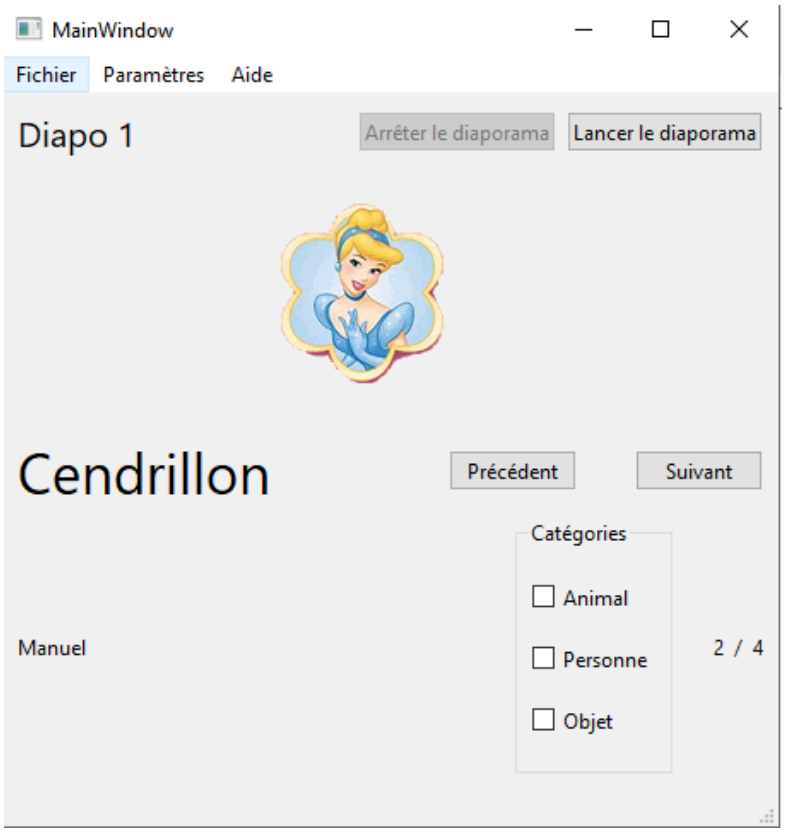
15.1 *Implémentation (v3)*

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas contient les méthodes permettant les interactions possibles sur les objets graphiques de l'interface du lecteur du 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. contient les méthodes permettant de réaliser les actions sur le lecteur de diaporama (avancer, reculer, …).
lecteur.cpp	Corps de la classe Lecteur.
image.h	Spécification de la classe Image contient les méthodes permettant de récupérer les informations d'une image (catégorie, rang, …).
image.cpp	Corps de la classe Image
main.cpp	Permet de lancer le projet.

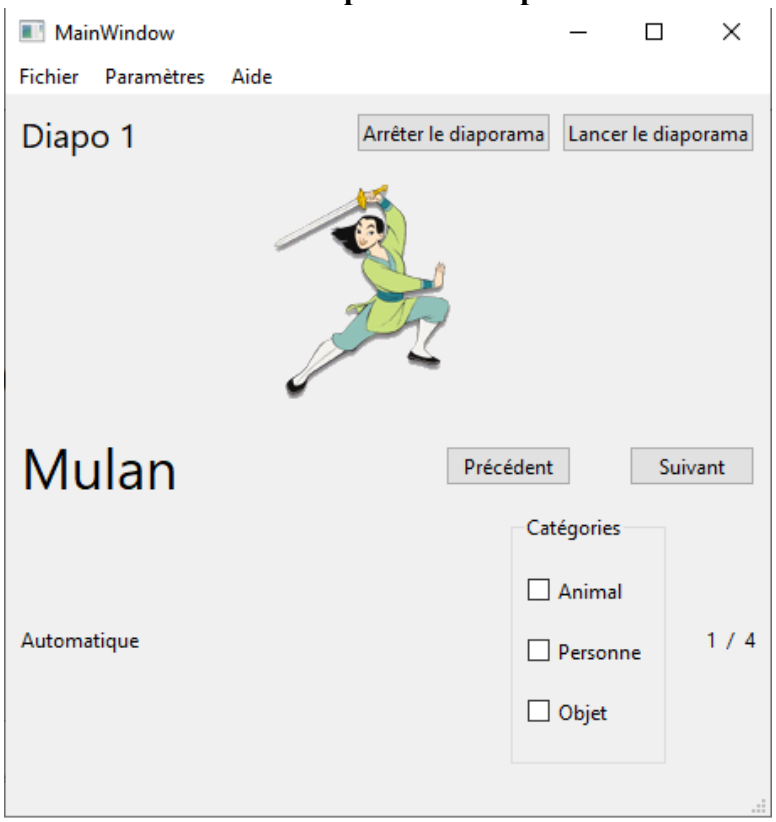
15.2 *Tests (v3)*

	Testeur	Yannis Duvignau		**Date**	05/06/2023
	Élement testé	Lecteur de diaporama V3		**Version**	1
	Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	
	Valide n°1	Passage du mode manuel à automatique	aucune	Le mode automatique se lance depuis la premiere image du diaporama	
	Valide n°2	Passage du mode automatique à manuel	aucune	Le mode automatique s'arrete et ple mode manuel recommence à l'image courrante	
	Valide n°3	Relancer le mode automatique(click bouton lancer diapo quand mode auto)	aucune	Le mode manuel se relance depuis la premiere image	
	Valide n°4	Une image s'affichent toutes les 5 secondes	aucune	Les images s'affichent bien les unes après les autres	
	Invalide n°1	Arret mode automatique quand mode manuel (click bouton arreter Diaporama)	aucune	Le bouton n'est pas cliquable en mode manuel	
	Invalide n°2	Enlever le diapo quand mode auto	aucune	Le bouton n'est pas cliquable en mode auto	

15.2.1 Passage du mode manuel à automatique

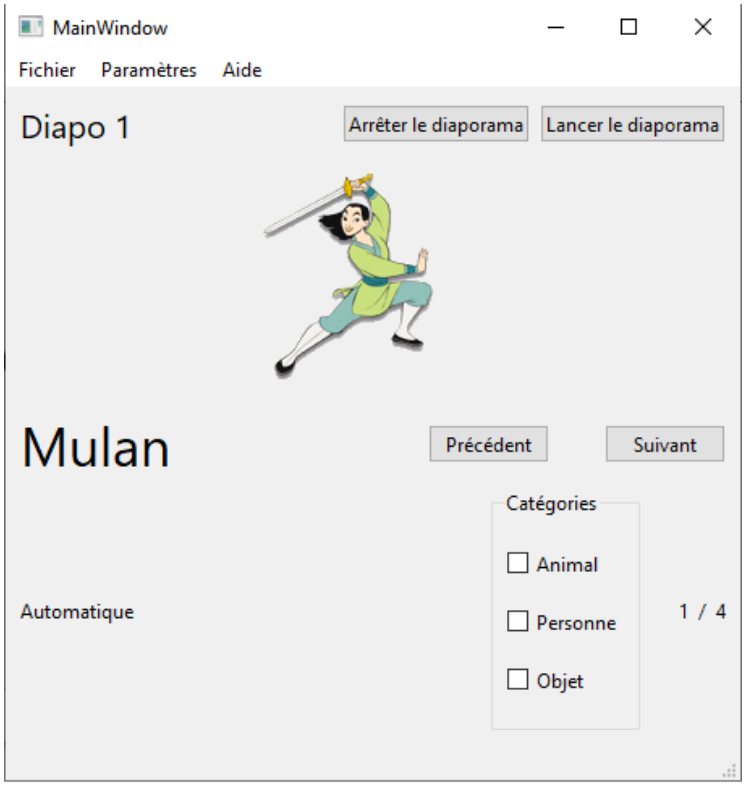


Le bouton “Lancer le diaporama” est pressé :

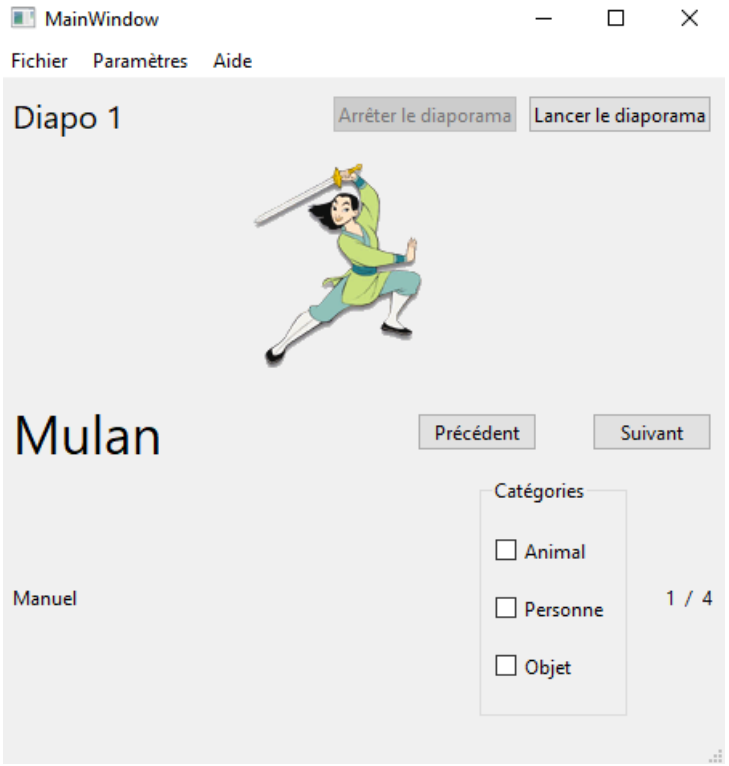


Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

15.2.2 Passage du mode automatique à manuel



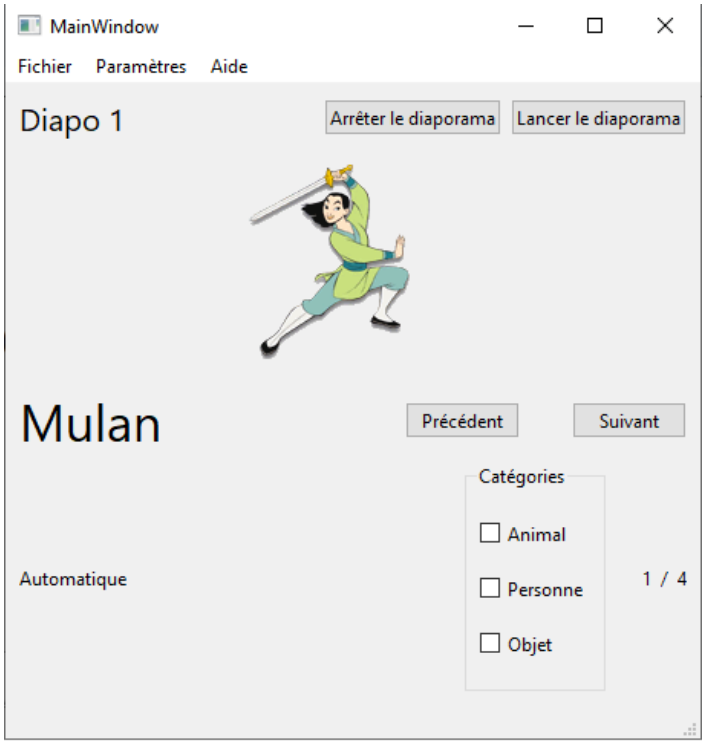
Le bouton “Arrêter le diaporama” est pressé :



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

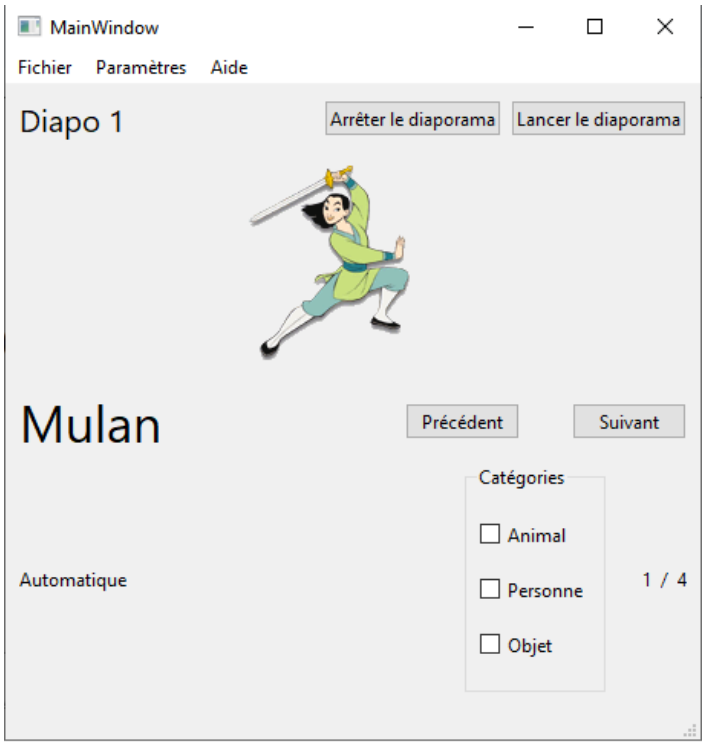
15.2.3 Relancer le mode automatique

Le bouton “Lancer le diaporama” est pressé :

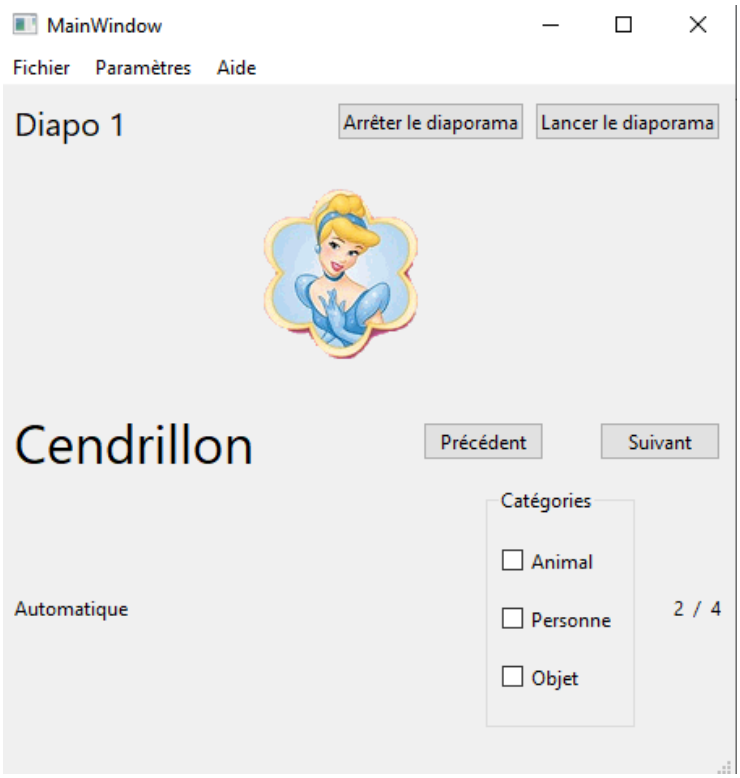


Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

15.2.4 Une image s’affiche toutes les 5 secondes



5 secondes après :

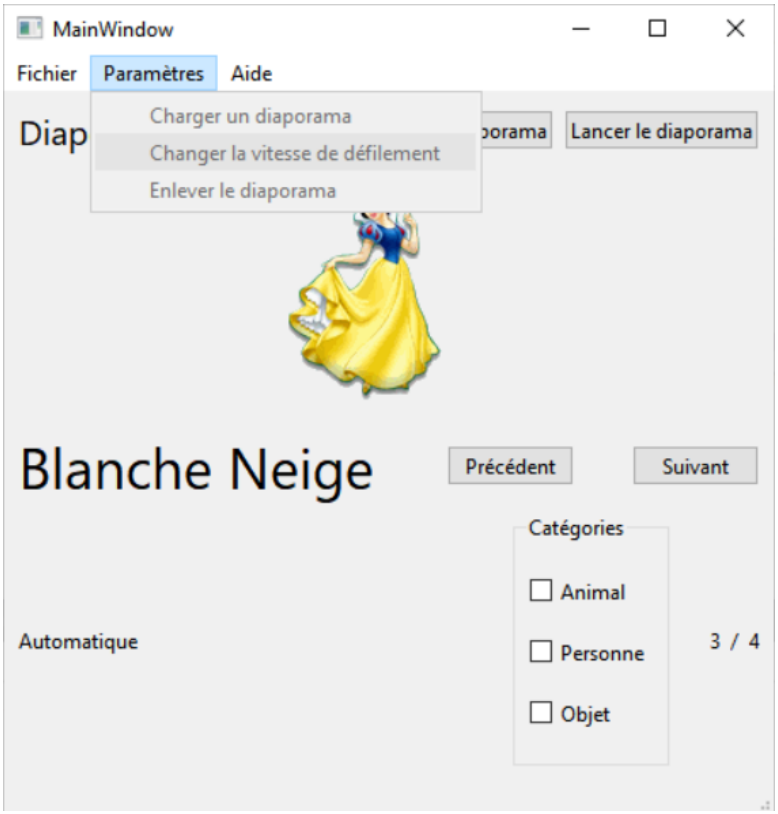


Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

15.2.5 Arrêt mode automatique en mode manuel

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

15.2.6 Enlever le diaporama en mode automatique



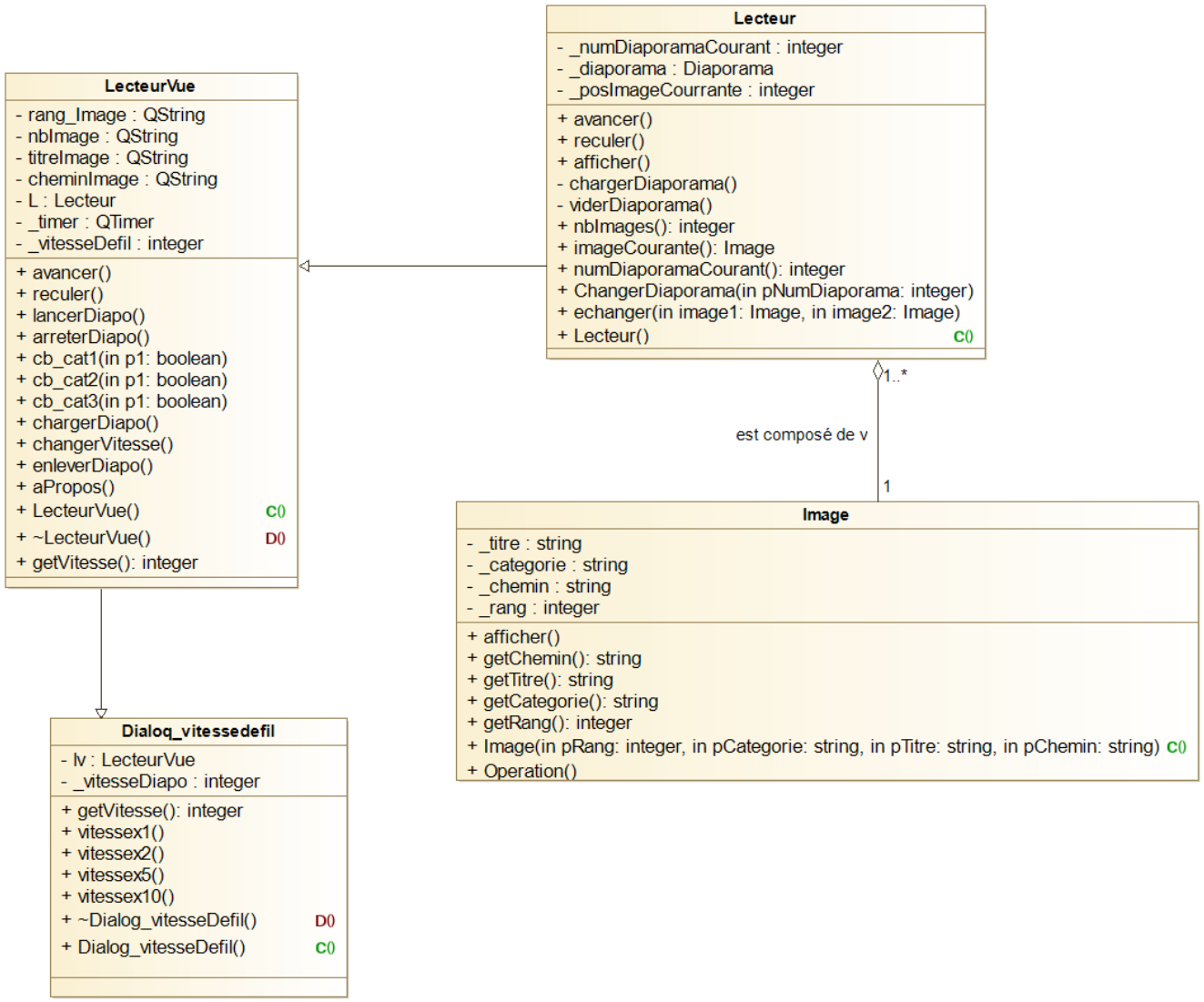
Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

Version v4 –

16. Description de la version

- Implémentation d'une boîte de dialogue permettant de modifier la vitesse de défilement des images
- Ajout du chargement des images en "dur" et les mets à disposition triées par ordre croissant de rang.
- Implémentation du paramètre qui enlève le diaporama en cours. Le Lecteur est vide à nouveau.

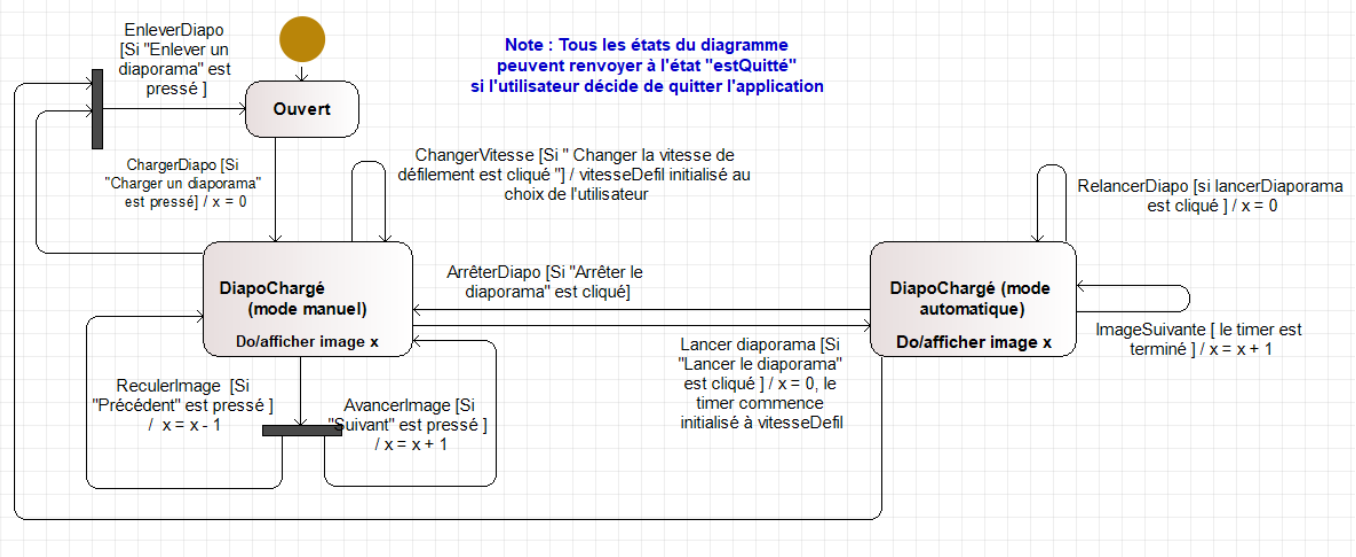
17. Diagramme de classe UML



(b) Dictionnaire des nouveaux éléments apportés pour chaque classe

Nom attribut	Signification	Type	Exemple	classe
`lv`	nouveau object LecteurVue	`LecteurVue`		`Dialog_vitessedefil`
`_vitesseDiapo`	vitesse du diaporama à changer	`QString`	`1000`	`Dialog_vitessedefil`

18. Diagramme d'état transitions



18.1 *Dictionnaire des états, événements et Actions (v4)*

Dictionnaire des états du diaporama

NomEtat	*Signification*
`Ouvert`	*Il est déclenché quand l'utilisateur ouvre le lecteur de diaporama. A ce stade, aucun diaporama n'est chargé.*
`DiapoChargé(mode manuel)`	*Un diaporama est chargé et l'utilisateur utilise le mode manuel (défilement des images à la main) de l'application.*

`DiapoChargé(mode automatique)`	*Un diaporama est chargé et l'utilisateur utilise le mode automatique (défilement automatique des images) de l'application.*

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`EnleverDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"quand l'utilisateur décide d'enlever le diaporama. Il renvoie à l'état "Ouvert".*
`ChargerDiapo`	*Cet événement est déclenché depuis l'état "Ouvert" quand l'utilisateur souhaite ouvrir un diaporama. Il renvoie à l'état "DiapoChargé".*
`LancerDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"(mode manuel) quand l'utilisateur souhaite faire défiler les images automatiquement. Il renvoie à l'état "DiapoChargé"(mode automatique).*
`ArreterDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"(mode automatique) quand l'utilisateur souhaite arrêter de faire défiler les images automatiquement. Il renvoie à l'état "DiapoChargé"(mode manuel).*
`RelancerDiapo`	*Cet événement est déclenché depuis l'état "DiapoChargé"(mode automatique) quand l'utilisateur souhaite relancer le diaporama. Il renvoie à l'état "DiapoChargé"(mode automatique).*
`ImageSuivante`	*Cet événement est déclenché depuis l'état "DiapoChargé"(mode automatique) quand le timer arrive à 0(timer terminé). Il renvoie à l'état "DiapoChargé"(mode automatique).*
`ReculerImage`	*Cet événement est déclenché quand l'utilisateur appuie sur le bouton "Précédent". L'image courante du diaporama est alors changée, puis affichée.*
`AvancerImage`	*Cet événement est déclenché quand l'utilisateur appuie sur le bouton "Suivant". L'image courante du diaporama est alors changée, puis affichée.*
`ChangerVitesse`	*Cet événement est déclenché depuis l'état "DiapoChargé"(mode manuel) quand l'utilisateur souhaite changer la vitesse de défilement pour le mode automatique. Il renvoie à l'état "DiapoChargé"(mode manuel).*

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*
`Afficher image x`	*Cette action est effectuée dans l'état "DiapoChargé". A chaque fois qu'une nouvelle image est chargée, elle est donc affichée.*

18.2 *Table T_EtatsEvenementsActions (v4)*

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 à	bouton enleverDiapo: a_enleverDiapo	bouton ChargerDiapo: a_chargerDiapo	bouton lancerDiapo:a_lancerDiapo	bouton arreterDiapo : b_arreterDiapo	bouton lancerDiapo a_lancer Diapo		bouton précédent : b_precedent	bouton suivant : b_suivant	bouton changementVitesse: a_changerVitesse
Évènement à *nomEtat*	EnleverDiapo	ChargerDiapo	LancerDiapo	ArreterDiapo	Relancer Diapo	ImageSuivante	ReculerImage	AvancerImage	ChangerVitesse
Ouvert		DiapoChargé(mode manuel), Chargement d'un diaporama							
DiapoChargé (mode manuel)	Ouvert, Enlèvement d'un diaporama		DiapoChargé, affichage de la première image du diaporama	DiapoChargé, affichage d'une nouvelle image			DiapoChargé, affichage d'une nouvelle image	DiapoChargé, affichage d'une nouvelle image	DiapoChargé, affichage d'une boîte de dialogue pour définir la vitesse de défilement
DiapoChargé (mode automatique)					DiapoChargé, affichage de la première image du diapora	DiapoChargé, affichage d'une nouvelle image			

					ma				

19. Implémentation et Tests

19.1 *Implémentation (v4)*

Liste et rôle des fichiers de cette version :

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas contient les méthodes permettant les interactions possibles sur les objets graphiques de l'interface du lecteur du 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. contient les méthodes permettant de réaliser les actions sur le lecteur de diaporama (avancer, reculer, …).
lecteur.cpp	Corps de la classe Lecteur.
image.h	Spécification de la classe Image contient les méthodes permettant de récupérer les informations d'une image (catégorie, rang, …).
image.cpp	Corps de la classe Image
main.cpp	Permet de lancer le projet.
dialog_vitessedefil.h	Spécification de la classe Dialog_vitesseDefil. contient les méthodes permettant de définir et de récupérer la vitesse de défilement du mode automatique.

dialog_vitessedefil.cpp	Corps de la classe Dialog_vitesseDefil
dialog_vitessedefil.ui	Fichier du dessin de l'interface réalisé par QtDesigner

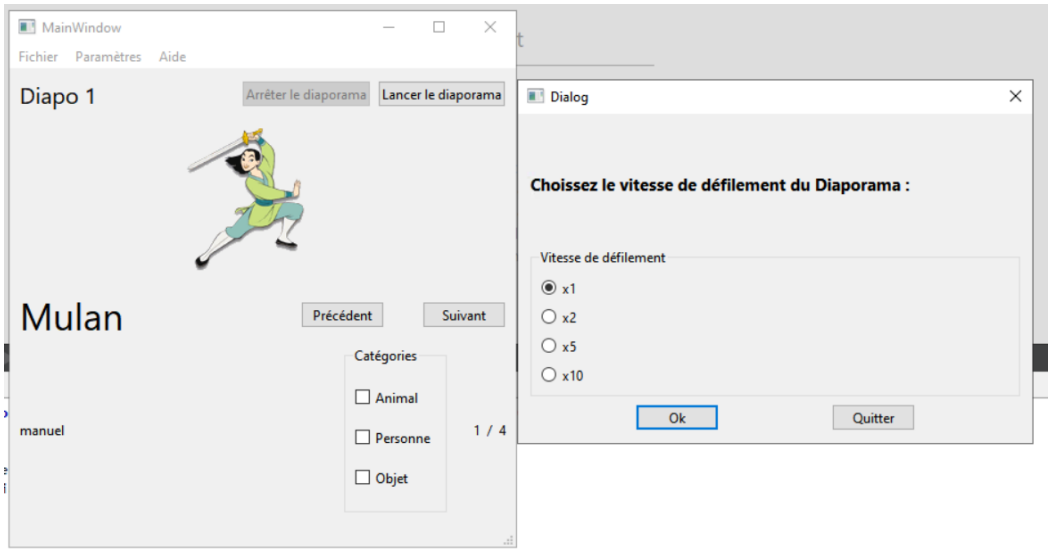
Remarques sur l'implémentation :

Dans la fenêtre de dialogue dialog_vitessedefil, nous avons fait le choix d'utiliser des boutons radio pour limiter les erreurs de saisie de l'utilisateur et conditionner les différentes vitesses de défilement possibles.

19.2 Tests (v4)

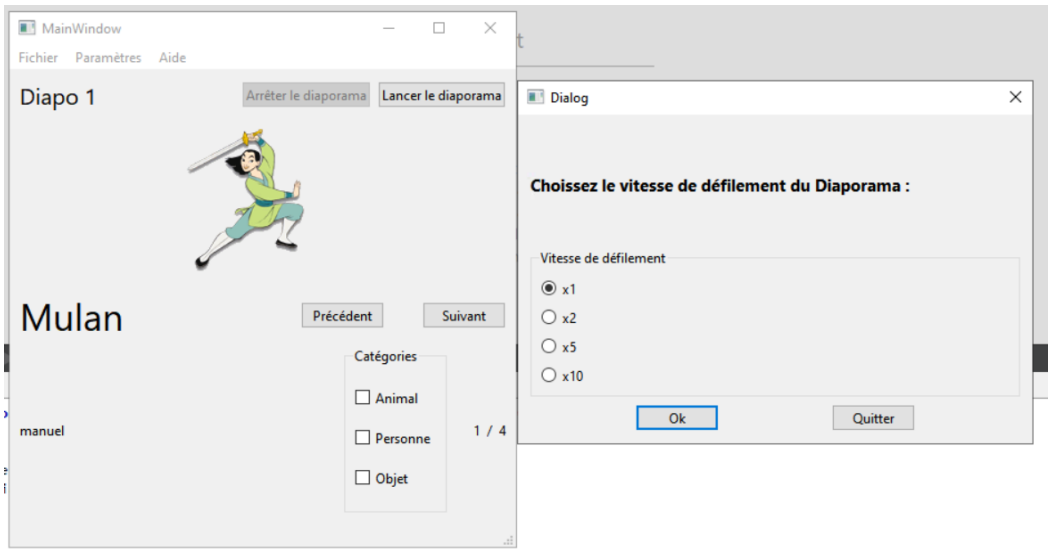
Testeur	Yannis Duvignau		**Date**	05/06/2023
Élement testé	Lecteur de diaporama V4		**Version**	1
Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	
Valide n°1	Changement de vitesse depuis boite de dialogue	aucune	vitesse changée	
Valide n°2	Annulation du changement de vitesse	aucune	vitesse pas changée	
Valide n°3	Charger un diaporama dans un Lecteur vide (sans diaporama chargé précédement)	aucune	chargement du diapo (affichage du diaporama et images)	
Valide n°4	Enlever le diaporama dans un Lecteur plein(avec diaporama chargé précédement)	aucune	Diapo enlevé (affichage du Lecteur sans diaporama chargé)	
Invalide n°1	Charger un diaporama dans un Lecteur plein (avec diaporama chargé précédement)	aucune	pas de chargement du diaporama	
Invalide n°2	Changement de vitesse en mode auto	aucune	Impossible de changer la vitesse	
Invalide n°3	Enlever le diaporama dans un Lecteur vide(sans diaporama chargé précédement)	aucune	Impossible d'enlever un diaporama	

19.2.1 Changement de vitesse depuis la boîte de dialogue



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

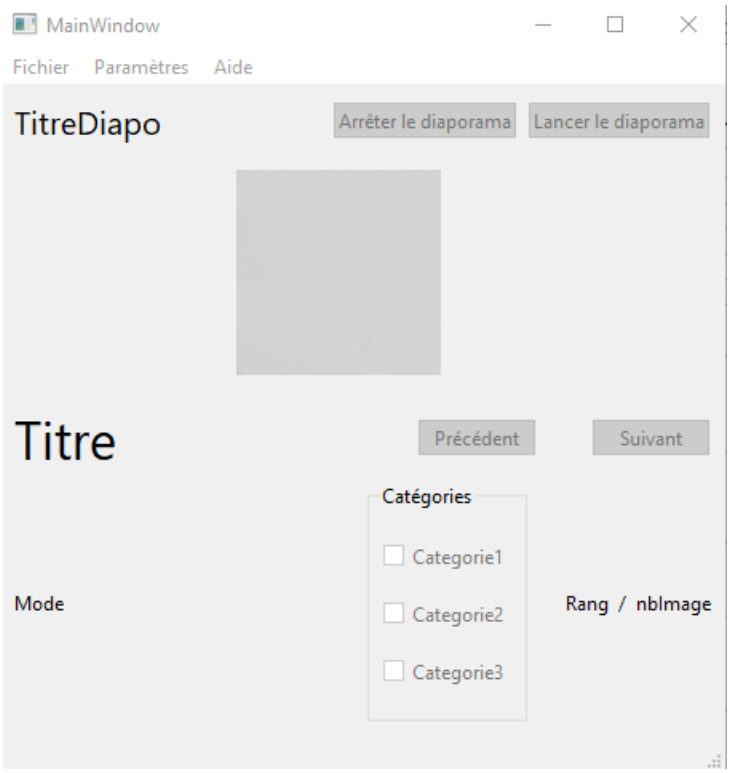
19.2.2 Annulation du changement de vitesse



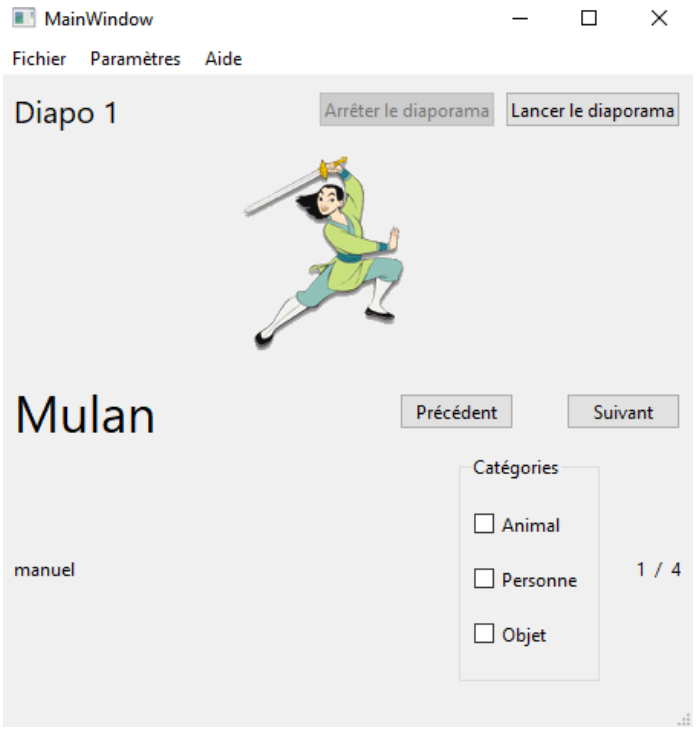
Bouton “Quitter” pressé :

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

19.2.3 Charger le diaporama dans un lecteur vide

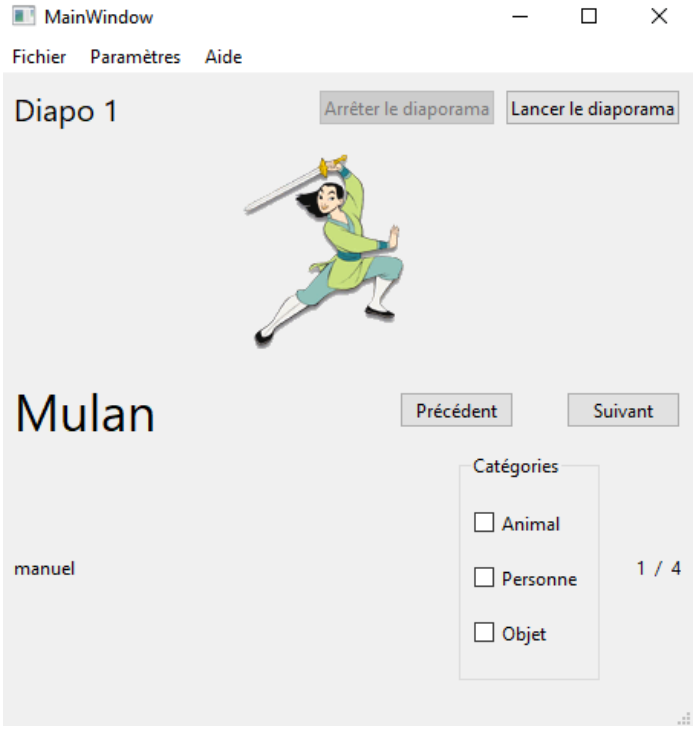


Bouton “Charger un diaporama” pressé :

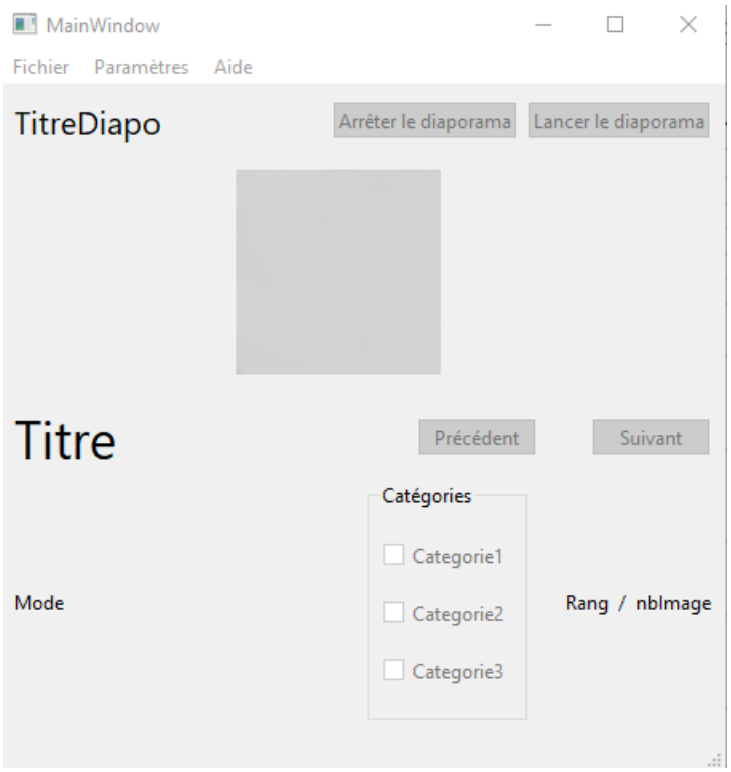


Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

19.2.4 Enlever le diaporama dans un lecteur plein



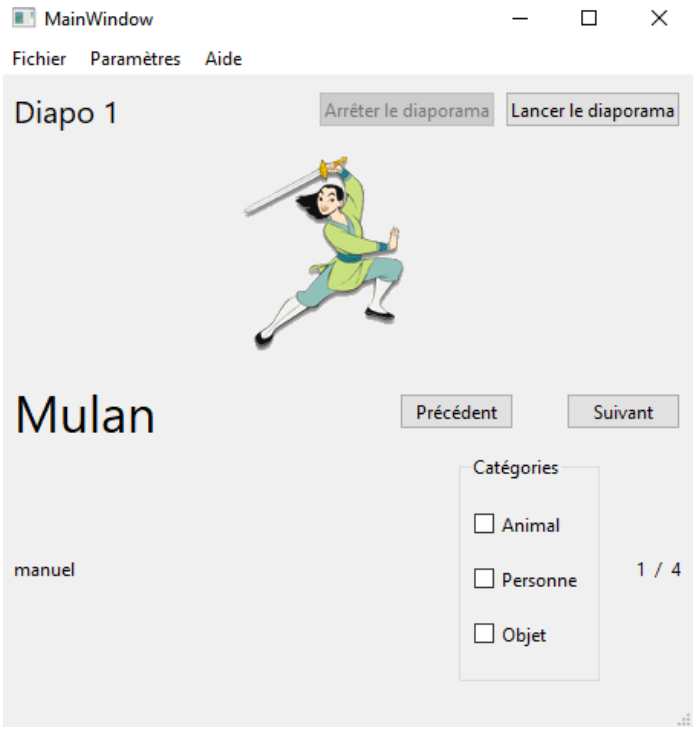
Bouton “Enlever le diaporama” pressé :



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

19.2.5 Charger un diaporama dans un lecteur plein

Bouton “Charger un diaporama” pressé :

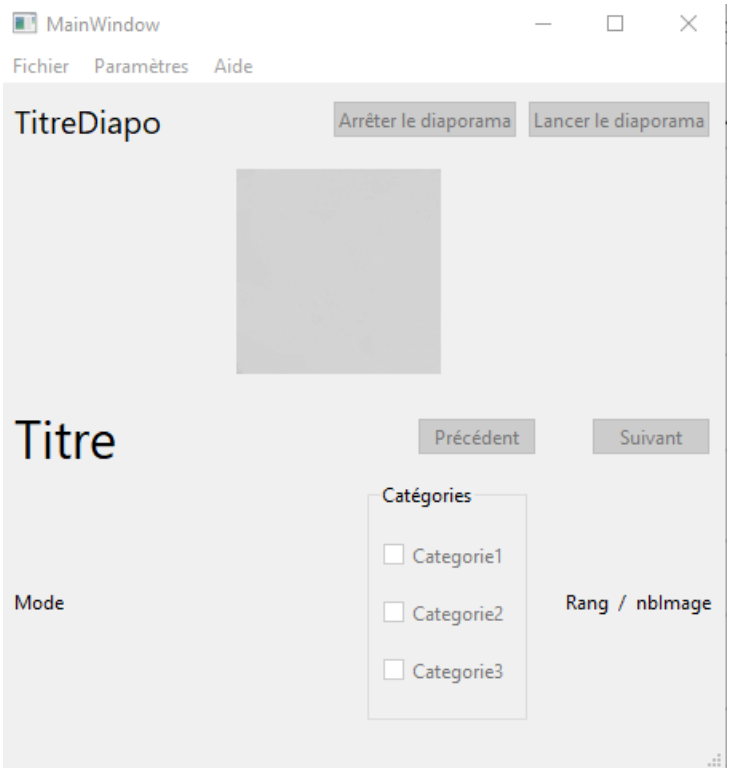


Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

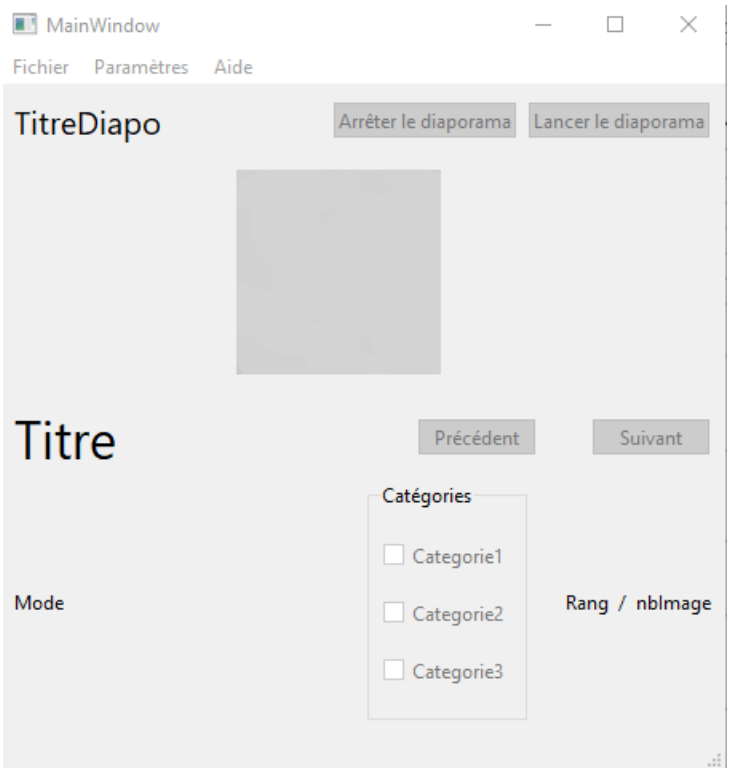
19.2.6 Changement de vitesse en mode automatique

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

19.2.7 Enlever le diaporama dans un lecteur vide



Bouton “Enlever le diaporama” pressé :



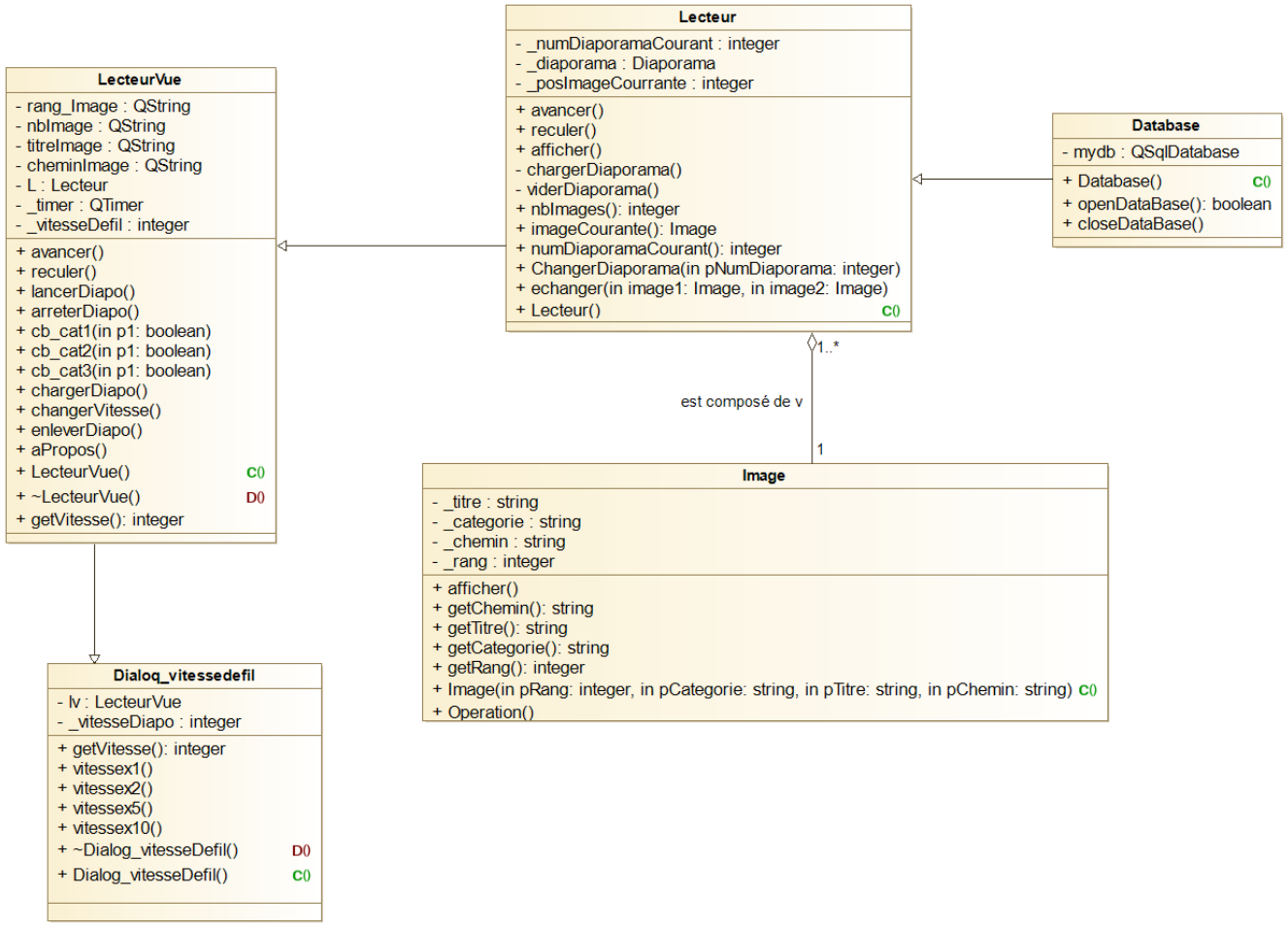
Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

Version v5 –

20. Description de la version

Exploitation des données de la base de données pour charger les informations relatives aux images depuis cette base de données. Il est important de noter que les informations relatives à la base de données sont seulement récupérées et ne sont pas encore exploitées (elles le seront dans la version 6 🙂).

21. Diagramme de classes (UML)

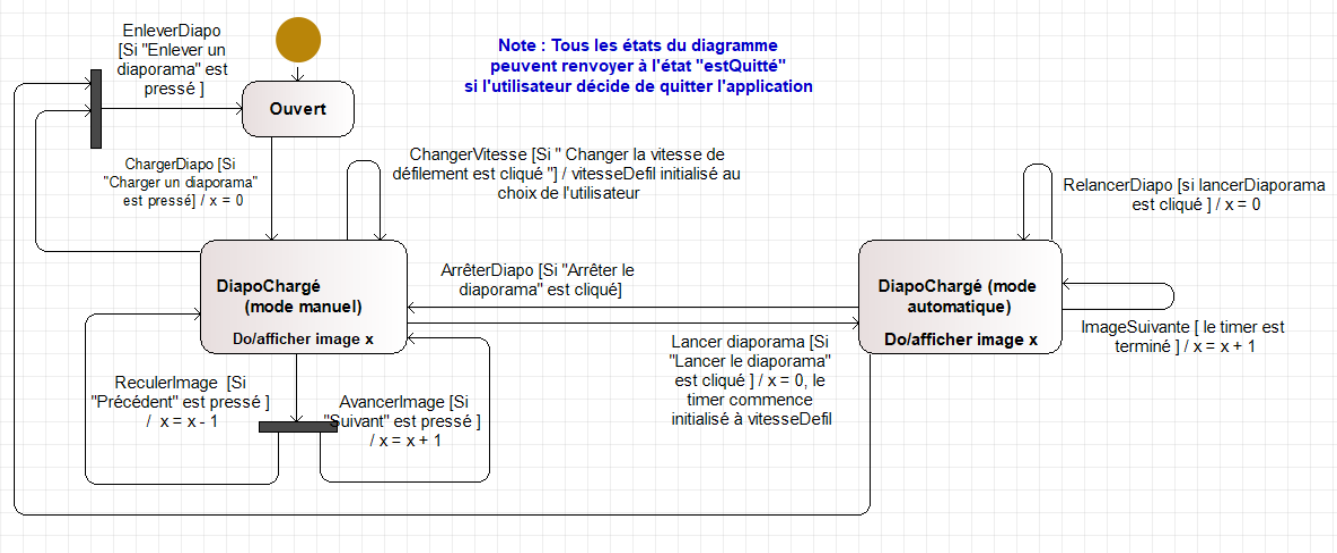


(b) Dictionnaire des nouveaux éléments apportés pour chaque classe

Nom attribut	Signification	Type	Exemple	classe
`mydb`	pointeur sur la base de donnée	`QSqlDatabase`		`Database`

22. Comportement de l'application

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



22.2 *Dictionnaire des états, événements et Actions (v5)*

Dictionnaire des états du diaporama

idem v4

Dictionnaire des événements faisant changer le diaporama d'état

idem v4

Description des actions réalisées lors de la traversée des transitions

idem v4

22.3 *Table T_EtatsEvenementsActions (v5)*

idem v4

23. Implémentation et tests

23.1 *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 contient les méthodes permettant les interactions possibles sur les objets graphiques de l'interface du lecteur du 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. contient les méthodes permettant de réaliser les actions sur le lecteur de diaporama (avancer, reculer, …).
lecteur.cpp	Corps de la classe Lecteur.
image.h	Spécification de la classe Image contient les méthodes permettant de récupérer les informations d'une image (catégorie, rang, …).
image.cpp	Corps de la classe Image
main.cpp	Permet de lancer le projet.
dialog_vitessedefil.h	Spécification de la classe Dialog_vitesseDefil. contient les méthodes permettant de définir et de récupérer la vitesse de défilement du mode automatique.
dialog_vitessedefil.cpp	Corps de la classe Dialog_vitesseDefil
dialog_vitessedefil.ui	Fichier du dessin de l'interface réalisé par QtDesigner

Remarques sur l'implémentation :

Idem V4

23.2 *Tests (v5)*

	Testeur	Yannis Duvignau		**Date**	05/06/2023
	Élement testé	Lecteur de diaporama V5		**Version**	1
	Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	
	Valide n°1	Connexion réussi avec la base de données	aucune	La connexion est faite avec la base de données	
	Valide n°2	La table existe	aucune	La table est présente dans la base de données	
	Valide n°3	Affichage des informations relatives aux images	aucune	Les informations s'affiche correctement	
	Invalide n°1	Echec connexion avec la base de données	aucune	La connexion n'est pas faite avec la base de données	
	Invalide n°2	La table n'existe pas	aucune	La table n'est pas présente dans la base de données	

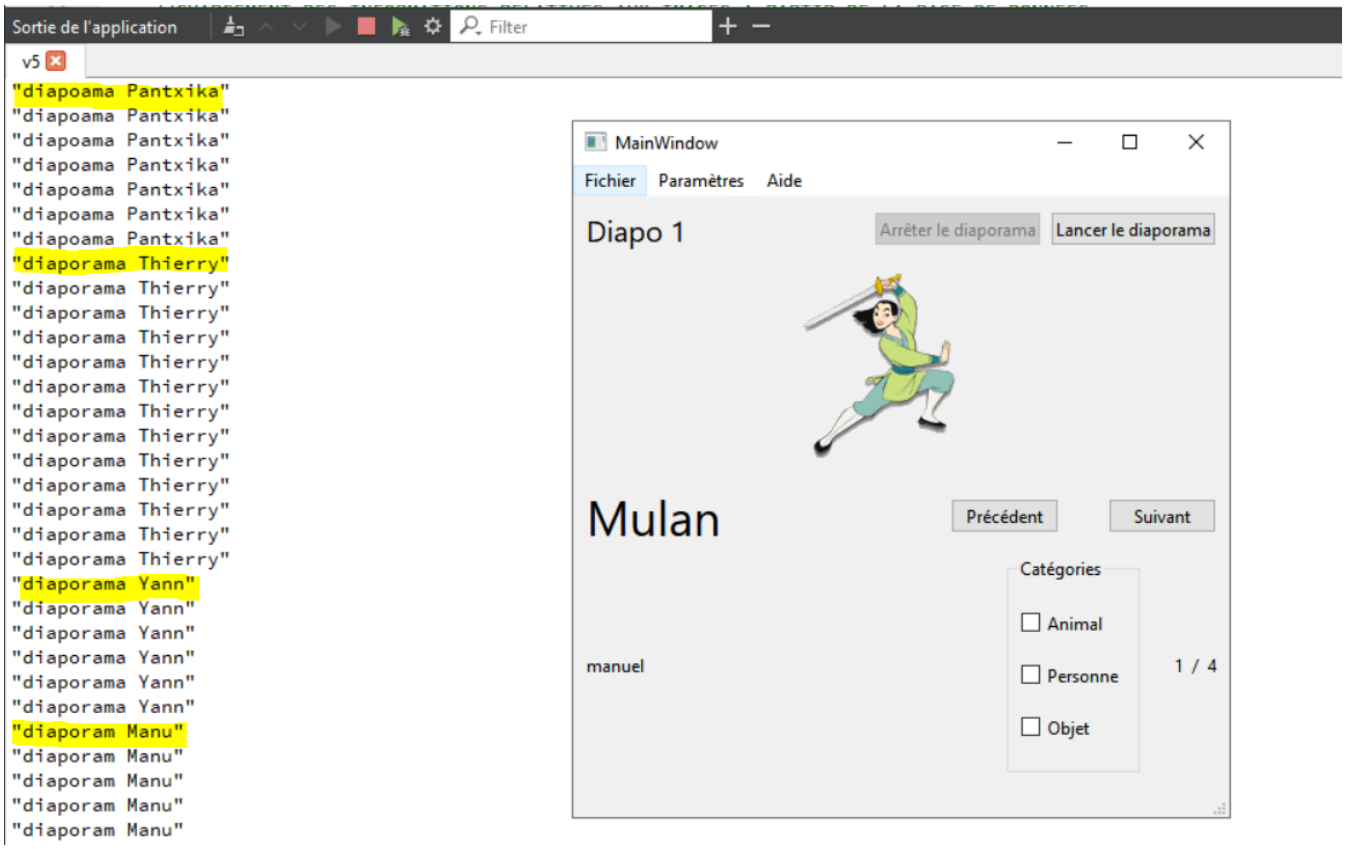
23.2.1 Connexion réussi avec la base de données

```
46      QSqlQuery Marequete;
47      Marequete.exec("SELECT idphoto, titrePhoto, uriPhoto, nomFamille, rang, `titre Diaporama`, vitesseDefilement "
48                     "FROM Diapos D "
49                     "JOIN DiaposDansDiaporama DDD ON D.idphoto = DDD.idDiapo "
50                     "JOIN Diaporamas ON DDD.idDiaporama = Diaporamas.idDiaporama "
51                     "JOIN Familles F ON D.idFam = F.idFamille ");
52
53      for (int i = 0; Marequete.next(); i++  )
54      {
55          int idphoto = Marequete.value(0).toInt();
56          QString titrePhoto = Marequete.value(1).toString();
57          QString uriPhoto = Marequete.value(2).toString();
58          QString nomFamille = Marequete.value(3).toString();
59          int rang = Marequete.value(4).toInt();
60          QString titre_Diaporama = Marequete.value(5).toString();
61          int vitesseDefilement = Marequete.value(6).toInt();
62
63          qDebug()<< idphoto;
64          qDebug()<< titrePhoto;
65          qDebug()<< uriPhoto;
66          qDebug()<< nomFamille;
67          qDebug()<< rang;
68          qDebug()<< titre_Diaporama;
69          qDebug()<< vitesseDefilement;
70
```

```
Sortie de l'application   Filter
v5
20:58:03: Starting \\haya\dossiersetud_BaieSsd\nconguisti\Downloads\S2.01-master\build-v5-Desktop_Qt_6_3_1_MinGW_64_bit-debugV5\debug\v5.exe...
changement du diaporama
oui ouvrir
0 images restantes dans le diaporama.
5
"Disney_4.gif"
"/cartesDisney/Disney_4.gif"
"Personnage"
7
"diapoama Pantxika"
1
8
"Disney_7.gif"
"/cartesDisney/Disney_7.gif"
```

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

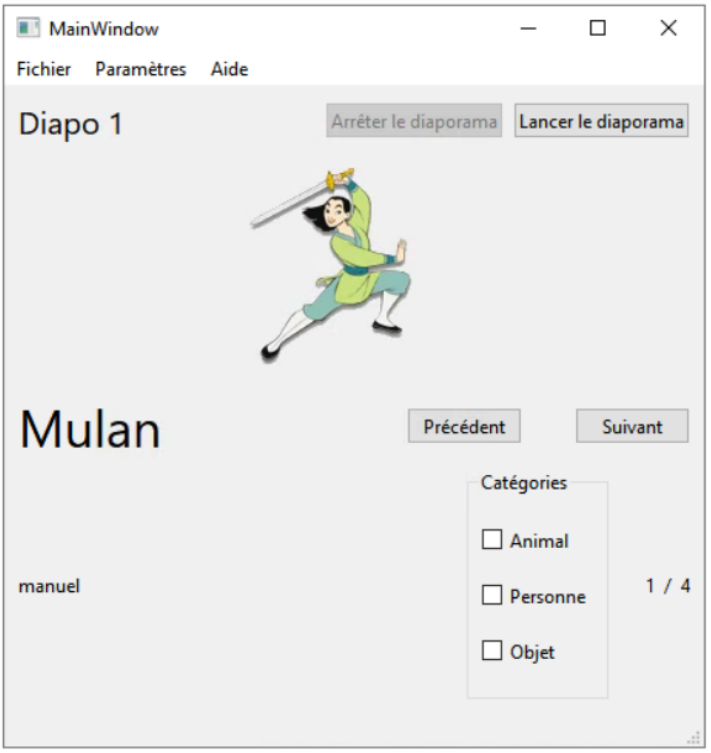
23.2.2 La table existe



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

23.2.3 Affichage des informations relatives aux images

```
Sortie de l'application                                  Filter
v5
"/cartesDisney/Disney_32.gif"
"Animal"
9
10
"Disney_9.gif"
"/cartesDisney/Disney_9.gif"
"Animal"
1
21
"Disney_20.gif"
"/cartesDisney/Disney_20.gif"
"Animal"
9
33
"Disney_32.gif"
"/cartesDisney/Disney_32.gif"
"Animal"
7
34
"Disney_33.gif"
"/cartesDisney/Disney_33.gif"
"Animal"
5
38
"Disney_37.gif"
"/cartesDisney/Disney_37.gif"
"Animal"
6
Diaporama num. 1 selectionne.
4 images chargees dans le diaporama
```



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

23.2.4 Echec de connexion avec la base de données

```
#define DATABASE_NAME "j'aime les coccinelles"
#define CONNECT_TYPE "QODBC"

Sortie de l'application                                  Filter
v5
21:42:28: Starting \\haya\dossiersetud_BaieSsd\nconguisti\Downloads\S2.01-master\build-v5-Desktop_Qt_6_3_1_MinGW_64_bit-debugV5\debug\v5.exe...
changement du diaporama
oui ouvrir
0 images restantes dans le diaporama.
QSqlQuery::exec: database not open
Diaporama num. 1 selectionne.
4 images chargees dans le diaporama
21:42:36: \\haya\dossiersetud_BaieSsd\nconguisti\Downloads\S2.01-master\build-v5-Desktop_Qt_6_3_1_MinGW_64_bit-debugV5\debug\v5.exe exited with code
0
```

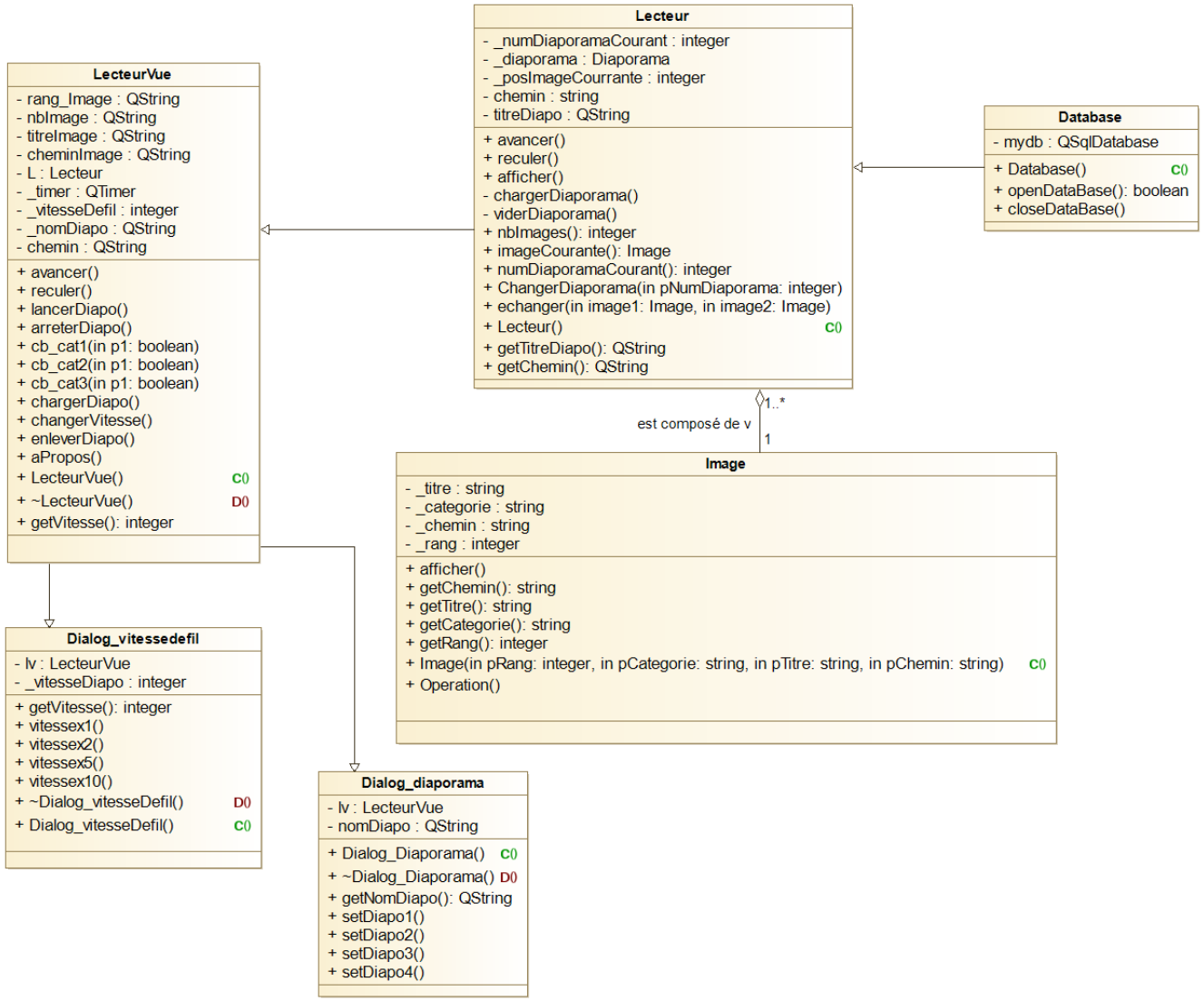
Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

Version 6 -

24. Description de la version

Ajout d'une Classe d'interface graphique Diaporama (Dialog_Diaporama) dans l'application. Nouvelle implémentation de Fichier >> Paramétrer >> Charger diaporama … : qui charge les informations relatives au Diaporama et aux Images à partir de la Bases de Données. => Exploitation de la base de données pour affichage des informations dans le lecteur.

25. Diagramme de classe UML

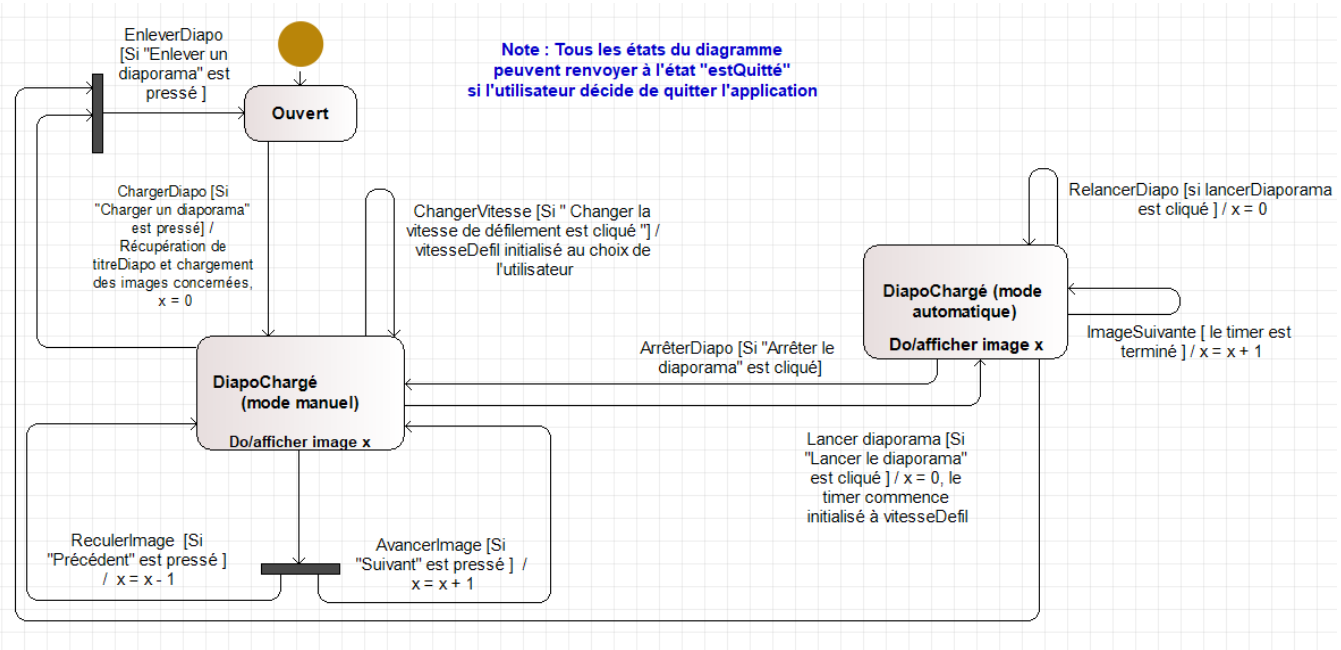


(b) Dictionnaire des nouveaux éléments apportés pour chaque classe

Nom attribut	Signification	Type	Exemple	classe
_nomDiapo	nom du diaporama courant	QString	Diapoama Pantxika	LecteurVue
chemin	chemin relatif de l'image courante	QString	F:/Documents/…	LecteurVue
titreDiapo	titre du diaporama à charger	QString	Diapoama Pantxika	Lecteur
chemin	chemin de l'image à charger	string	/cartesDisney/popol.gif	Lecteur
lv	nouveau object LecteurVue	LecteurVue		Dialog_Diaporama
nomDiapo	choix de l'utilisateur du diaporama à charger	QString	Diapoama Pantxika	Dialog_Diaporama

26. Comportement de l'application

26.1 *Diagramme états-transitions-actions du lecteur de diaporamas (v6)*



26.2 *Dictionnaire des états, événements et Actions (v6)*

Dictionnaire des états du diaporama

idem que v4

Dictionnaire des événements faisant changer le diaporama d'état

idem que v4

Description des actions réalisées lors de la traversée des transitions

idem que v4

26.3 *Table T_EtatsEvenementsActions (v6)*

Élément graphique prenant en charge cet événement à	bouton enleverDiapo: a_enleverDiapo	bouton ChargerDiapo: a_chargerDiapo	bouton lancerDiapo: a_lancerDiapo	bouton arreterDiapo: b_arreterDiapo	bouton lancerDiapo a_lancerDiapo		bouton précédent : b_precedent	bouton suivant : b_suivant	bouton changementVitesse: a_changerVitesse
Évènement à *nomEtat*	EnleverDiapo	ChargerDiapo	LancerDiapo	ArreterDiapo	RelancerDiapo	ImageSuivante	ReculerImage	AvancerImage	ChangerVitesse
Ouvert		DiapoChargé(mode manuel), Chargement d'un diaporama choisi par l'utilisateur.							
DiapoChargé (mode manuel)	Ouvert, Enlèvement d'un diaporama		DiapoChargé, affichage de la première image du diaporama	DiapoChargé, affichage d'une nouvelle image			DiapoChargé, affichage d'une nouvelle image	DiapoChargé, affichage d'une nouvelle image	DiapoChargé, affichage d'une boîte de dialogue pour définir la vitesse de défilement
DiapoChargé (mode automatique)					DiapoChargé, affichage de la première image du diaporama	DiapoChargé, affichage d'une nouvelle image			

27. Implémentation et tests

27.1 Implémentation (v6)

Liste et rôle des fichiers de cette version :

lecteurVue.h	Spécification de la classe graphique Qt contenant l'interface du lecteur de diaporamas contient les méthodes permettant les interactions possibles sur les objets graphiques de l'interface du lecteur du 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. contient les méthodes permettant de réaliser les actions sur le lecteur de diaporama (avancer, reculer, …).
lecteur.cpp	Corps de la classe Lecteur.
image.h	Spécification de la classe Image contient les méthodes permettant de récupérer les informations d'une image (catégorie, rang, …).
image.cpp	Corps de la classe Image
main.cpp	Permet de lancer le projet.
dialog_vitessedefil.h	Spécification de la classe Dialog_vitesseDefil. contient les méthodes permettant de définir et de récupérer la vitesse de défilement du mode automatique.
dialog_vitessedefil.cpp	Corps de la classe Dialog_vitesseDefil

dialog_vitessedefil.ui	Fichier du dessin de l'interface réalisé par QtDesigner
dialog_diaporama.h	Spécification de la classe Dialog_Diaporama. contient les méthodes permettant de définir et de récupérer le diaporama choisi et à charger par l'utilisateur
dialog_diaporama.cpp	Corps de la classe Dialog_Diaporama
dialog_diaporama.ui	Fichier du dessin de l'interface réalisé par QtDesigner

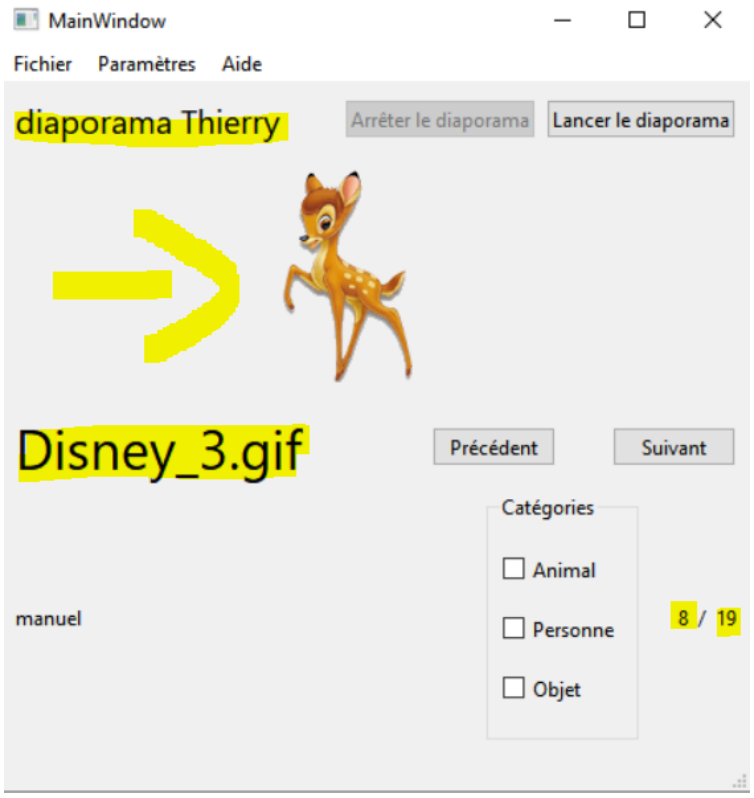
Remarques sur l'implémentation :

Lors de l'ajout de la fenêtre de dialogue dialog_diaporama, nous avons fait le choix d'implémenter des boutons radio car on ne doit pouvoir choisir qu'un seul diaporama parmi 4 choix possibles.

27.2 Tests

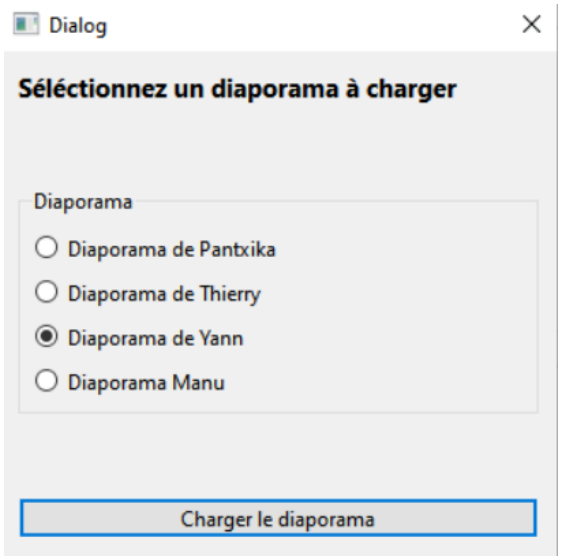
Testeur	Yannis Duvignau		**Date**	05/06/2023
Élement testé	Lecteur de diaporama V6		**Version**	1
Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	
Valide n°1	Elements graphique correction récupérés et affichés	aucune	affichage correcte des caractéristiques des images	
Valide n°2	Choix diaporama et cohérence affichage (les images affiché sont les images du diaporama chargé)	aucune	Le diaporama choisi est celui afficher	
Invalide n°1	Aucune diaporama selectionner mais chargé quand même	aucune	Fermer fenetre de dialogue et reste dans aucun diaporama chargé (état "Ouvert"	
Invalide n°2	Charger un diaporama quand lecteur plein (un diaporama est déjà chargé)	aucune	Impossible de charger un autre diaporama	

27.2.1 Éléments graphiques correction récupérés et affichés



Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

27.2.2 Choix diaporama et cohérence affichage

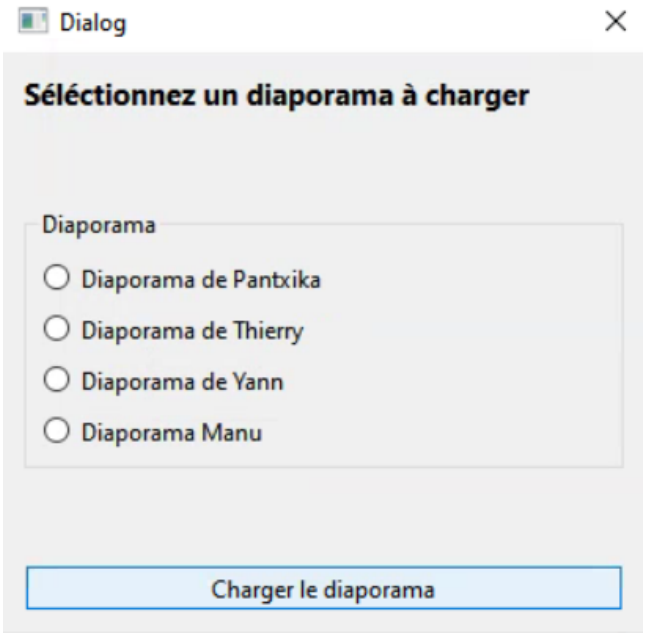


Bouton “Charger le diaporama” pressé :

Seul les images du diaporama de Yann sont affichées

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

27.2.3 Aucun diaporama sélectionné mais chargé quand même

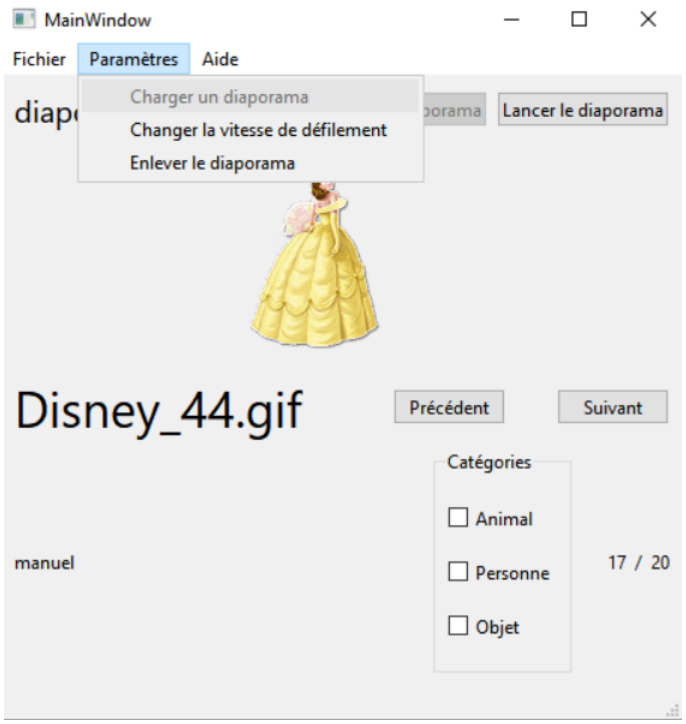


Bouton “Charger le diaporama” pressé : Si aucun diaporama n’a été cliqué, le programme s'arrête.

```
21:53:05: \\haya\dossiersetud_BaieSsd\nconguisti\Documents\IUT\Annee1\S2\S2.01\S2.01-parfaite\v6\debug\v6.exe crashed.
```

Les résultats obtenus ne sont pas conformes aux attentes, l'application se ferme subitement alors qu'elle était censée revenir à la fenêtre principale.

27.2.4 Charger un diaporama quand lecteur plein (un diaporama est déjà chargé)



On ne peut pas charger un diaporama quand un autre diaporama est chargé.

Les résultats obtenus sont conformes aux résultats attendus. Le test est donc validé.

Bilan

Dépôt Git où trouver le projet complet (les versions réalisées) :

Le lien vers le dépôt Git est le suivant : dydyvicto/S2.01 (github.com)

Utilisation du chemin d'accès selon l'utilisateur :

V1 à V5 : Utilisation d'un fichier "ressources".

V6 : Utilisation d'une variable "chemin" située ligne 54 de Lecteur.cpp (à modifier).

Temps global de travail (pour le groupe) :

Le temps global de travail sur le projet est de 54 heures.

En moyenne chacun à passé 10 heures sur le projet

Apprentissages majeurs :

- Dossier ressources pour améliorer le code est la destination des images
- Connexion avec la base de données pour affichage des images
- Rédaction de livrables de conception d'une application
- Développement d'une application complète en programmation objet c++
- Une utilisation fructueuse de l'outil de versionnage de code "github". 🙂

Difficultés majeures :

- Affichage des informations avec base de données
- Filtre par catégorie (pas encore implémenté)
- Dans la documentation, diagramme état-transition et tests

Points positifs / négatifs de l'activité ;

Positifs :

- application des notions vus en cours et en TD/TP pour fabrication d'une vraie application
- sujet intéressant

Négatifs :

- sujet un peu vague
- Livrable de conception un peu confu.