Université de Pau et des Pays de l'Adour - IUT DE BAYONNE - PAYS-BASQUE année 2022-2023
Département Informatique saé 20.1

Saé 2.01 – Développement d'une application

Lecteur de diaporamas – Dossier d'Analyse et conception

---

1. Compléments de spécifications externes.

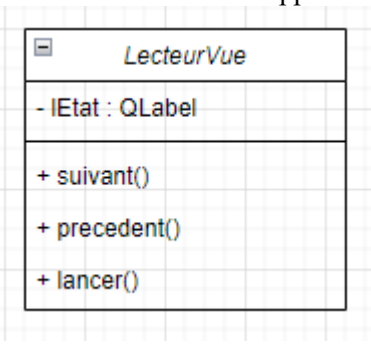
On précise uniquement les points qui vous ont semblé flous ou bien incomplets. Rien de plus à signaler dans cette étude.

2. Scénarios

	Enchaînement Nominal	
Messages	**Acteur : Utilisateur**	**Système : Lecteur de diaporama**
1	L'utilisateur lance l'application	
2		Le système démarre en mode Manuel et affiche la barre de statut
3	L'utilisateur charge le diaporama souhaité qu'il choisis dans la BD	
4		Le système affiche le diaporama choisis en commençant par la première image
5	L'utilisateur navigue dans le diaporama et va à l'image suivante	
6		Le système affiche l'image suivante
7	L'utilisateur filtre les images par catégorie avec l'option correspondante	
8		Le système filtre les images comme souhaité par l'utilisateur
9	L'utilisateur supprime le diaporama en cours de visualisation	
10		Le système décharge le diaporama
11	L'utilisateur quitte l'application	
	Enchaînements Alternatifs	
	Acteur : Utilisateur	**Système : Lecteur de diaporama**
	Mode Auto	
4.A	L'utilisateur choisit le mode Auto	
5.A		Le système passe le diaporama en mode Auto et fait défiler le diaporama à raison d'une image par secondes
6.A	Passage au message 9	

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 XXXXX			
Nom attribut	**Signification**	**Type**	**Exemple**
`lEtat`	Label etat et un label obervant l'état d'une image	`QLabel`	`image`
`suivant()`	méthodes pour passer à la diapo suivante	`void`	
`precedent()`	méthodes pour passer à la diapo précédent	`void`	
`lancer()`	méthodes pour lancer le diaporama	`void`	

Tableau 2 : Dictionnaire des éléments - Classe xxx

(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 – 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 fournit main.cpp

Valeurs fournies:

- images chargées: 4
- avancé 4 fois
- reculer 5 fois

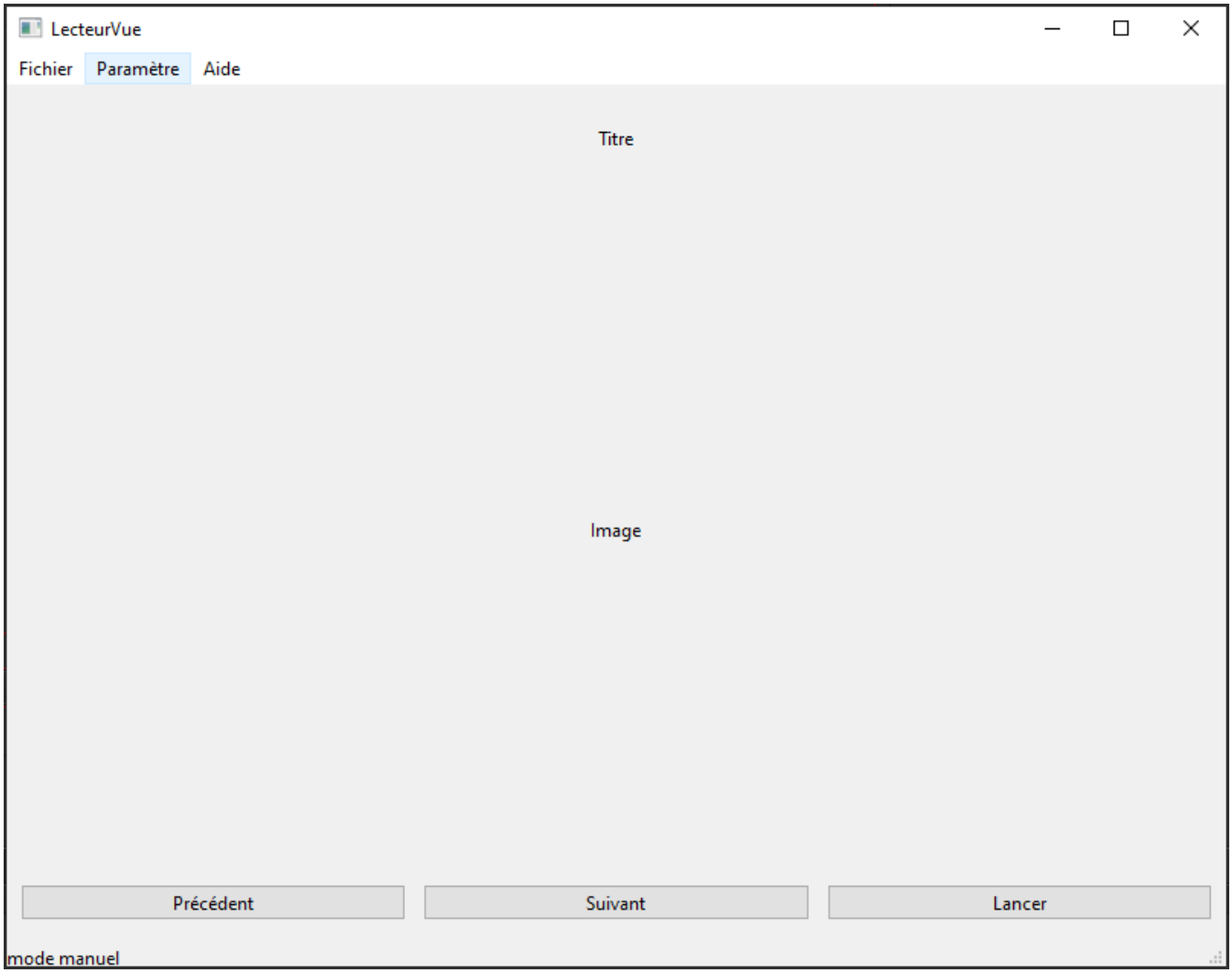
Valeurs attendues:

- charge 4 images avec en première courante “grincheux”
- avance 4 fois pour revenir à l’image “grincheux”
- recule 5 fois pour finir sur l’image “mickey”

```
Diaporama num. 1 selectionne.
4 images chargees dans le diaporama
Lecteur du diaporama num. 1
Image courante :
image( rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)

Test avancer() : 4 fois
avancer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)
avancer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
avancer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)
avancer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)

Test reculer() : 5 fois
reculer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)
reculer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:3, titre:Blanche Neige, categorie:personne, chemin:C:\cartesDisney\carteDisney2.gif)
reculer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:2, titre:Cendrillon, categorie:personne, chemin:C:\cartesDisney\carteDisney4.gif)
reculer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:1, titre:Grincheux, categorie:personne, chemin:C:\cartesDisney\carteDisney1.gif)
reculer() :
Lecteur du diaporama num. 1
Image courante :
image( rang:4, titre:Mickey, categorie:animal, chemin:C:\cartesDisney\carteDisney1.gif)

Enlever le diaporama courant = Choisir diaporama 0
```

Version v1 – projet Graphique seul

5. Éléments d'interface



Éléments	nom	position
Titre	lTitre	en haut de la page au centre
image	lImage	au centre de la page
Bouton Précédent	bPrecedent	en bas à gauche
Bouton Suivant	bSuivant	en bas au centre de l'écran
Bouton Lancer	bLancer	en bas à droite de l'écran
Paramètre	menuParam_tre	2 ème position de la tool bar

mode manuel	lEtat	Dans la bar de status

6. 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 :

```
connect(ui->bLancer, SIGNAL(clicked()), this, SLOT(lancer()));
connect(ui->bSuivant, SIGNAL(clicked()), this, SLOT(suivant()));
connect(ui->bPrecedent, SIGNAL(clicked()), this, SLOT(precedent()));
```

6.2 Test

-

Slots	Description	Résultat(s) attendu(s)	Résultat(s) obtenu(s)
avancer()	Affiche un message lorsque le bouton *Suivant* est cliqué	"Vous avez cliqué sur le bouton avancer."	"Vous avez cliqué sur le bouton avancer."
reculer()	Affiche un message lorsque le bouton *Précédent* est cliqué	"Vous avez cliqué sur le bouton reculer."	"Vous avez cliqué sur le bouton reculer."
lancer()	Affiche un message lorsque le bouton *Lancer Diaporama* est cliqué	"Vous avez cliqué sur le bouton démarrer le diapo."	"Vous avez cliqué sur le bouton démarrer le diapo."

Version v2 –

7. Diagramme de classes (UML)

A faire – s'il y a des changements - sinon indiquer que idem v0

8. Comportement de l'application

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

A faire

Figure 9 : Diagramme états-transitions du lecteur de diaporamas – v2

8.2 Dictionnaire des états, événements et Actions (v2)

Dictionnaire des états du diaporama

nomEtat	*Signification*

Tableau 2 : États du lecteur de diaporamas – v2

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*

Tableau 3 : Evénements faisant changer le diaporama d'état – v2

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*

Tableau 4 : Actions à réaliser lors des changements d'état – lecteur de diaporamas v2

8.3 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 pregnant en charge cet événement →			
Événement → *nomEtat*			

Tableau 5 : Matrice d'états-transitions du lecteur de diaporamas – v2

L'intérêt de cette vue matricielle est qu'elle permet une préparation naturelle et aisée de l'étape suivante de programmation.

9. Implémentation et tests

9.1 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 Préciser le rôle
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. Préciser le rôle
lecteur.cpp	Corps de la classe Lecteur
image.h	Spécification de la classe Image Préciser le rôle
image.cpp	Corps de la classe Image
main.cpp	??

Remarques sur l'implémentation :

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

9.2 Tests (v2)

Classe	Description	Valeur(s) en entrée	Résultat(s) attendu(s)	Résultat(s) obtenu(s)
valide n°1	Le chemin d'accès correspond à une image d'extension .gif	"C:\\cartesDisney\\carteDisney_0.gif"	Affichage de l'image dans l'interface	Affichage de l'image dans l'interface
valide n°2	La catégorie de l'image est contenu dans les valeurs possibles ("personne", "animal" ou "objet")	"animal"	Affichage de la catégorie dans l'interface	Affichage de la catégorie dans l'interface
valide n°3	Le chemin d'accès correspond à une image d'extension .png	"C:\\cartesDisney\\carteDisney_0.png"	Affichage de l'image dans l'interface	Affichage de l'image dans l'interface
valide n°4	L'utilisateur clique sur le bouton *Suivant*	Le bouton *pbSuivant* est activé	Défilement à l'image suivante et affichage des informations dans la console	Défilement à l'image suivante et affichage des informations dans la console
valide n°5	L'utilisateur clique sur le bouton *Précédent*	Le bouton *pbPrecedent* est activé	Retour à l'image précédente et affichage des informations dans la console	Retour à l'image précédente et affichage des informations dans la console
valide n°6	L'utilisateur clique sur le bouton *Démarrer Diaporama*	Le bouton *pbToggleAuto* est activé	Changement du statut en mode automatique et affichage d'un message dans la console	Changement du statut en mode automatique et affichage d'un message dans la console
valide n°7	L'utilisateur clique sur le bouton *Arrêter Diaporama*	Le bouton *pbToggleAuto* est activé	Changement du statut du diaporama en mode manuel et affichage d'un message dans la console	Changement du statut du diaporama en mode manuel et affichage d'un message dans la console
valide n°8	L'utilisateur clique sur le bouton *Quitter*	L'action *actionSortir* est appelée	Fermeture de la fenêtre	Fermeture de la fenêtre
valide n°9	L'utilisateur clique sur le bouton *A propos de*	L'action *action_propos_de* est appelée	Affichage d'une fenêtre contenant la version de l'application, la date de création et les auteurs	Affichage d'une fenêtre contenant la version de l'application, la date de création et les auteurs
valide n°10	L'utilisateur clique sur le bouton *Vitesse de défilement*	L'action *actionChanger_diaporama* est appelée	Affichage d'un message dans la console	Affichage d'un message dans la console
valide n°11	L'utilisateur clique sur le bouton *Enlever le diaporama*	L'action *actionEnlever_diaporama* est appelée	Affichage d'un message dans la console et vide le diaporama	Affichage d'un message dans la console et vide le diaporama

valide n°12	L'utilisateur clique sur le bouton *Charger le diaporama*	L'action *actionChanger_diaporama* est appelée	Affichage d'un message dans la console et changement du diaporama	Affichage d'un message dans la console et changement du diaporama
invalide n°1	Le chemin d'accès ne correspond à aucune image	"C:\\cartesDisney\\carteDisney_70.gif"	Echec	Echec
invalide n°2	Le chemin d'accès correspond à un fichier texte	"C:\\cartesDisney\\carteDisney_0.txt"	Echec	Echec
invalide n°3	Le chemin d'accès est vide	""	Echec	Echec
invalide n°4	La catégorie de l'image n'est pas contenue dans les valeurs possibles	"nourriture"	Echec	Echec
invalide n°5	La catégorie de l'image est vide	""	Echec	Echec
invalide n°6	Le titre de l'image est vide	""	Echec	Echec

Version v5 –

10. Diagramme de classes (UML)

A faire – s'il y a des changements - sinon indiquer que idem vXX

11. Comportement de l'application

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

A faire

Figure 9 : Diagramme états-transitions du lecteur de diaporamas – v5

11.2 Dictionnaire des états, événements et Actions (v5)

Dictionnaire des états du diaporama

nomEtat	*Signification*

Tableau 2 : États du lecteur de diaporamas – v5

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*

Tableau 3 : Evénements faisant changer le diaporama d'état – v5

Description des actions réalisées lors de la traversée des transitions

nomAction	*Signification*

Tableau 4 : 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 pregnant en charge cet événement →			
Événement → *nomEtat*			

Tableau 5 : Matrice d'états-transitions du lecteur de diaporamas – v5

L'intérêt de cette vue matricielle est qu'elle permet une préparation naturelle et aisée de l'étape suivante de programmation.

12. 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 Préciser le rôle
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 Préciser le rôle
lecteur.cpp	Corps de la classe Lecteur
image.h	Spécification de la classe Image Préciser le rôle
image.cpp	Corps de la classe Image
main.cpp	??

Remarques sur l'implémentation :

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

12.2 Tests (v5)

A faire :

Décrire les tests prévus / réalisés pour montrer :
- Le comportement de l'interface non lié aux aspects fonctionnels du programme
- Le comportement de l'interface liée aux aspects fonctionnels du programme
- Le comportement fonctionnel de l'application

13. Bilan

Dépôt Git où trouver le projet complet (les versions réalisées):
https://github.com/LOVE-Keqing/LecteurDiaporama

Temps global de travail (pour le groupe):
Nous avons tous travaillé pendant les séances ainsi qu'une partie chez nous.

Apprentissages majeurs:
Nous avons appris à utiliser GitHub et approfondi nos connaissances et notre maîtrise de Qt.

Difficultés majeures:
Nous avons eu quelques difficultés au début avec GitHub mais nous avons reçu beaucoup d'aide de la part de Esteban, membre d'un autre groupe.
Il y a également eu quelques difficultés avec les premiers tests, pour qu'ils soient intéressants et pertinents.

Points positifs / négatifs de l'activité:
Nous pensons que le sujet ainsi que les consignes sur comment faire le rapport étaient mal expliquées, pour certaines choses, nous ne savions pas exactement comment le faire, et il a fallu demander plusieurs fois de l'aide aux professeurs ou à d'autres groupes.
Il y avait une bonne entente dans le groupe et nous avons donc pus collaborer efficacement.