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

Compléments de spécifications externes.

l’application devra pouvoir faire un filtre;filtrant les images du diaporama par catégorie : “personnage”,”animal”,”objet”,”tous”,ce mode a été rajouté pour pouvoir avoir le diaporama sans filtre.
Ceci va être appliqué grâce à un bouton qui permettra de switcher entre les modes. Tu appuie sur le bouton sa change de mode.

Scénarios

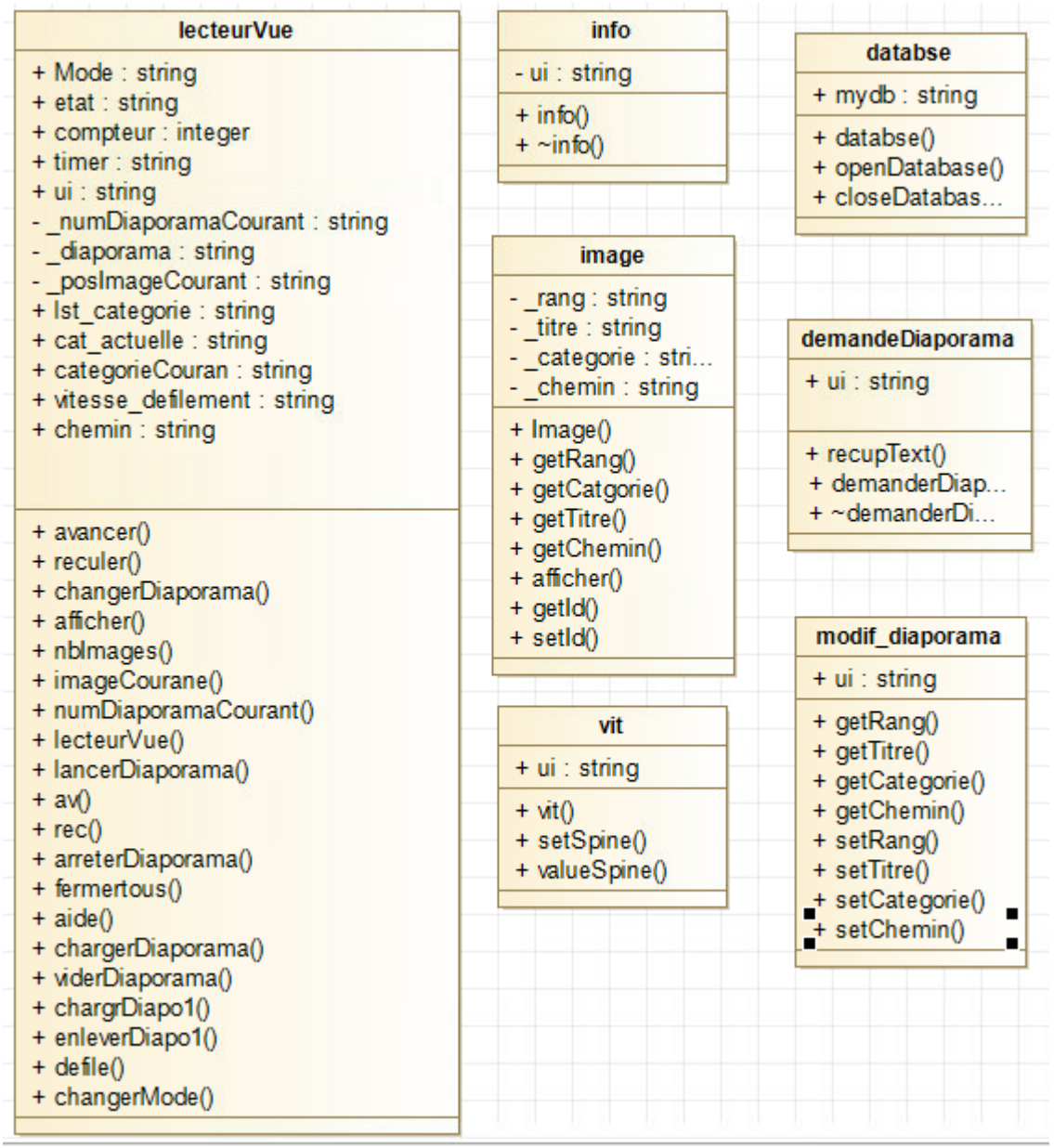
scénario lancer diaporama

Titre	lancer le diaporama		
Acteurs	utilisateur		
Résumé	permet de lancer le diaporama n mode automatique		
Fiche	Description réelle		
Métadonnée	Création : 16/05/2023 modifiée le : 16/05/2023 version 1		
Précondition	être sur l'application		
Postcondition			
Messages	**Acteur : utilisateur**	**Système : l' application QT**	**Maquette**
1		Le système affiche la page principale	***page principale***
	l'utilisateur demande a charger un diaporama		***page principale***
2		le système montre au client un espace pour indiquer quelle diaporama il veut	***page principale***
3	le client indique quelle diaporama il veut		***page principale + demande de diaporama***
		le système charge le diaporama demander et affiche la première image .	***page principale***
	le client demande afficher l'image précédente		***page principale***
		le système passe le diaporama en mdoe manuelle puis affiche l'image précédente(il affiche fait une boucle si on est a la première)	***page principale***
	le client demande a changer le filtre du diaporama		***page principale***
		le système changera le filtre passant au suivant	***page principale***
	le client demande a lancer le diaproama		***page principale***
		le systeme passera le diaporama en automatique puis passera a l'image suivante toute les X suivante (X etant présciser quand le diaporama est charger)	***page principale***
	le client demande de l'aide		***page principale + aide***
		le système affiche les informations de l'application	***page principale + aide***

		le systeme affiche les informations de l'application	page d'aide
	le client demande a changer la vitesse de défilement		*page principale + modification de la vitesse de défilement*
		le système ouvre un espace ou le client peut exprimer la vitesse voulu	*page principale + modification de la vitesse de défilement*
	le client remplie l'espace et valide		*page principale + modification de la vitesse de défilement*
		le système enregistre les changements pour le diaporama charger (pas en ligne juste sur le moment)	*page principale*
	le client demande a modifier limage actuel		*page principale + modification de diapo*
		le systeme montre au client un espace pour que le client puisse modifier les informations de l'image (ces modification seront modifier dans la base de donner)	*page principale + modification de diapo*
	le client demande a modifier une diapositive		*page principale + modification de diapo*
		le systeme montre au client un endroit pour changer les information de la diapositive actuel	*page principale + modification de diapo*
	le client remplie et valide		*page principale + modification de diapo*
		le systeme applique les modifications a la base de donner	*page principale + modification de diapo*

Diagramme de classe (UML)

Le diagramme de classes UML



Dictionnaire des éléments

Classe database

Nom attribut	**Signification**	**Type**	**Exemple**
`mydb`	la variable qui représente la liaison avec la base de données	`QSqlDatabase`	

Classe demandeDiaporama			
Nom attribut	**Signification**	**Type**	**Exemple**
`ui`	le pointeur vers la partie graphique	`Ui`	

Classe modif_diaporama			
Nom attribut	**Signification**	**Type**	**Exemple**
`ui`	le pointeur vers la partie graphique	`Ui`	

Classe image			
Nom attribut	**Signification**	**Type**	**Exemple**
`rang`	*rang de l'image au sein du diaporama auquel l'image est associée*	`unsigned int`	`1`
`_titre`	*intitulé de l'image*	`string`	`"disney.git"`
`_categorie`	*catégorie de l'image (personne, animal, objet)*	`string`	`"objet"`

_chemin	*chemin complet vers le dossier où se trouve l'image/ pas a afficher*	string	"/carte/disney.git"

Classe info			
Nom attribut	**Signification**	**Type**	**Exemple**
ui	le pointeur vers la partie graphique	Ui	

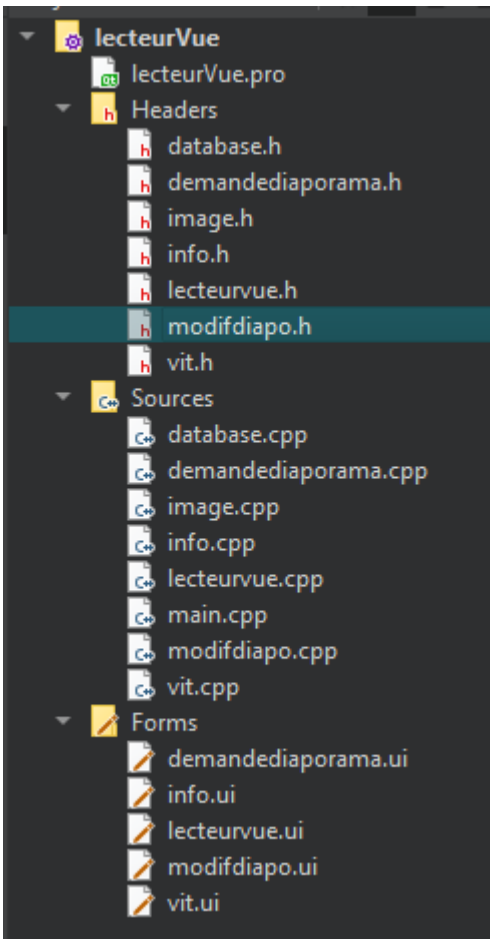
Classe lecteurVue			
Nom attribut	**Signification**	**Type**	**Exemple**
Mode	l'enum listant les modes de l'application	enum	
lst_categorie[4]	la liste contenant les different catégorie d'image possible	string	
cat_actuelle	c'est un compteur ou indicateur pour la catégorie actuelle	int	0
CategorieImageCourant	la variable contenant la categorie d'image actuelle pour le diaporama	string	"tous"
etat	la variable indiquant l'état de l'application	Mode	manuel
compteur	un compteur permettant de compter les seconde pour faire défiler les images	int	0
timer	l'objet QTimer permettant de faire le timer	QTimer	
vitesse_defilement	l'entier qui indique la vitesse de défilement du diaporama	int	2
db	le pointeur vers la base de donner	database	

chemin	la chaine de caractère du chemin pour les images	string	F:\\ecole\\SAE\\SAE S2.01

Classe vit			
Nom attribut	**Signification**	**Type**	**Exemple**
ui	le pointeur vers l'élément graphique	Ui	1

different dossiers

liste des dossiers



Liste et rôle des fichiers de cette version

database.h	spécification de la classe databse
demandediaporama.h	spécification de la classe demandediaporama
image.h	Spécification de la classe Image
info.h	Spécification de la classe info
lecteurVue.h	Spécification de la classe lecteurVue
modif_diaporama.h	Spécification de la classe modif_diaporama
vit.h	Spécification de la classe vit
modifdiapo.h	spécification de la classe modifdiapo
database.cpp	Corps de la classe database
demandeDiaporama.cpp	Corps de la classe demandeDiaporama
Image.cpp	Corps de la classe database
info.cpp	Corps de la classe database
lecteurVue.cpp	Corps de la classe lecteurVue
modif_diaporama.cpp	Corps de la classe modif_diaporama

modifdiapo.cpp	corps de la classe modifdiapo
vit.cpp	Corps de la classe vit
main.cpp	Teste les méthodes de la classe Lecteur

signal est SLOT (dans lecteurVue.h)

```
public slots:
    void lancerDiaporama();
    void av();//avancer
    void rec();//reculer
    void arreterDiaporama();//arrete le diaporama
    void fermertous();//ferme tous
    void aide();//affiche la fenetre
    void chargerdiapo1();//chager le diaporama 1
    void enleverdiapo1();
    void defile();
    void changerMode();
    void modifier_Diapositive();
```

Les fichiers

entêtes

database

```h
#ifndef DATABASE_H
#define DATABASE_H

#include <QSqlDatabase>

#define DATABASE_NAME "nodenot_bd9"
#define CONNECT_TYPE "QODBC"

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

private:
    QSqlDatabase mydb;
};

#endif // DATABASE_H
```

demandeDiaporama

```h
#ifndef DEMANDEDIAPORAMA_H
#define DEMANDEDIAPORAMA_H

#include <QDialog>

namespace Ui {
class demandeDiaporama;
}

class demandeDiaporama : public QDialog
{
    Q_OBJECT

public:
    explicit demandeDiaporama(QWidget *parent = nullptr);
    ~demandeDiaporama();
    QString recupTexte();

private:
    Ui::demandeDiaporama *ui;
};

#endif // DEMANDEDIAPORAMA_H
```

image

```h
#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();

    string getId ();
    void setId(string);

    void setRang(unsigned int);
    void setCategorie(string);
    void setTitre(string);
    void setChemin(string);
    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/ pas a afficher
    string Id;
};

#endif // IMAGE_H
```

info

```h
#ifndef INFO_H
#define INFO_H

#include <QDialog>

namespace Ui {
class info;
}

class info : public QDialog
{
    Q_OBJECT

public:
    explicit info(QWidget *parent = nullptr);
    ~info();

private:
    Ui::info *ui;
};

#endif // INFO_H
```

lecteurVue

```h
#ifndef LECTEURVUE_H
#define LECTEURVUE_H
#include "image.h"
#include <QMainWindow>
#include "info.h"
#include "vit.h"
#include <vector>
#include <QTimer>
#include "database.h"
#include "demandediaporama.h"

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

QT_BEGIN_NAMESPACE
namespace Ui { class lecteurVue; }
QT_END_NAMESPACE

class lecteurVue : public QMainWindow
{
    Q_OBJECT

public:
    enum Mode {automatique,manuel};
    string lst_categorie[4] = {"tous","Objet","Animal","Personnage"};
    int cat_actuelle=0;
    string CategorieImageCourant="tous";
    Mode etat;
    int compteur=0;
    QTimer timer;
    int vitesse_defilement=2;
    database *db;
    string chemin = "F:\\ecole\\SAE\\SAE S2.01\\projet-avec-git-S2.01\\cartesDisney";

    void avancer(int);              // incrémente _posImageCourante, modulo nbImages()
    void reculer(int);              // décrémente _posImageCourante, modulo nbImages()
    void changerDiaporama(unsigned int pNumDiaporama);    // permet de choisir un dia
    void afficher();            // affiche les informations sur lecteur-diaporama et
    unsigned int nbImages();    // affiche la taille de _diaporama
    Image* imageCourante();     // retourne le pointeur vers l'image courante
    unsigned int numDiaporamaCourant();
    lecteurVue(QWidget *parent = nullptr);
    ~lecteurVue();
    void lancer();
```

```
public slots:
    void lancerDiaporama();
    void av();//avancer
    void rec();//reculer
    void arreterDiaporama();//arrete le diaporama
    void fermertous();//ferme tous
    void aide();//affiche la fenetre
    void chargerdiapo1();//chager le diaporama 1
    void enleverdiapo1();
    void defile();
    void changerMode();


private:
    Ui::lecteurVue *ui;

    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 _numDiaporama
    void viderDiaporama();      // vide _diaporama de tous ses objets image et les de


};
#endif // LECTEURVUE_H
```

modif_diaporama

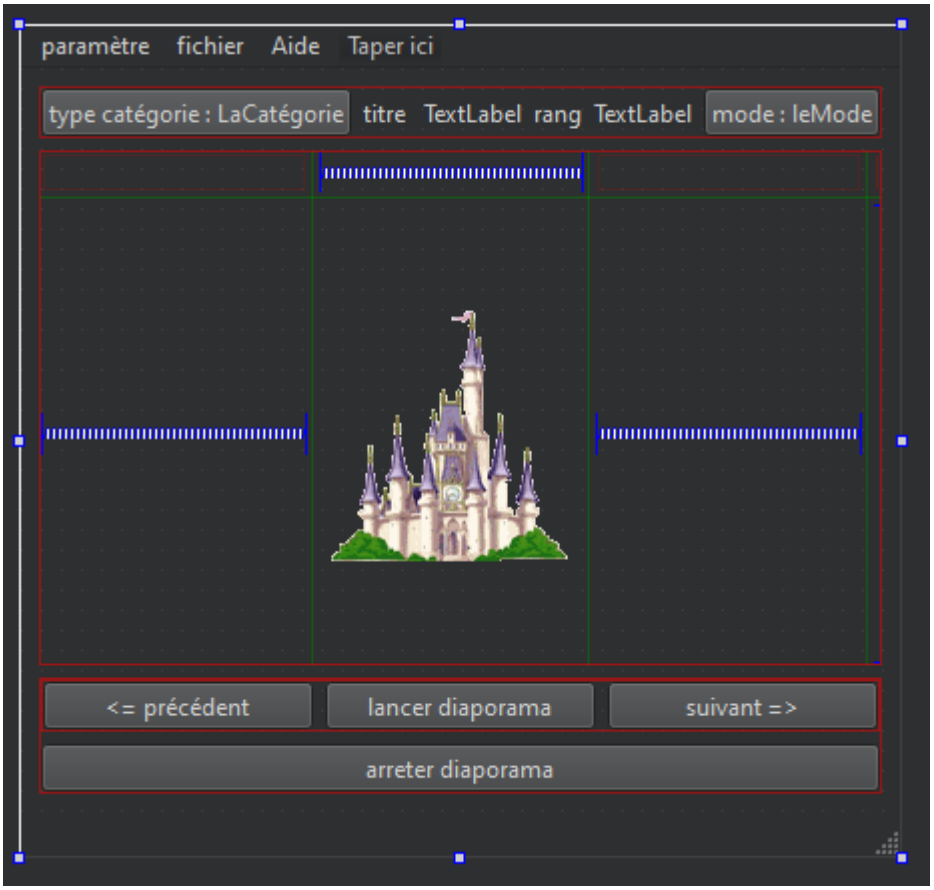
```
#ifndef MODIFDIAPO_H
#define MODIFDIAPO_H

#include <QDialog>
using namespace std;

namespace Ui {
class modifDiapo;
}

class modifDiapo : public QDialog
{
    Q_OBJECT

public:
    explicit modifDiapo(QWidget *parent = nullptr);
    ~modifDiapo();
    void setRang(QString);
    void setTitre(QString);
    void setCategorie(QString);
    void setChemin(QString);

    QString getRang();
    QString getTitre();
    QString getCategorie();
    QString getChemin();


private:
    Ui::modifDiapo *ui;
};

#endif // MODIFDIAPO_H
```

vit

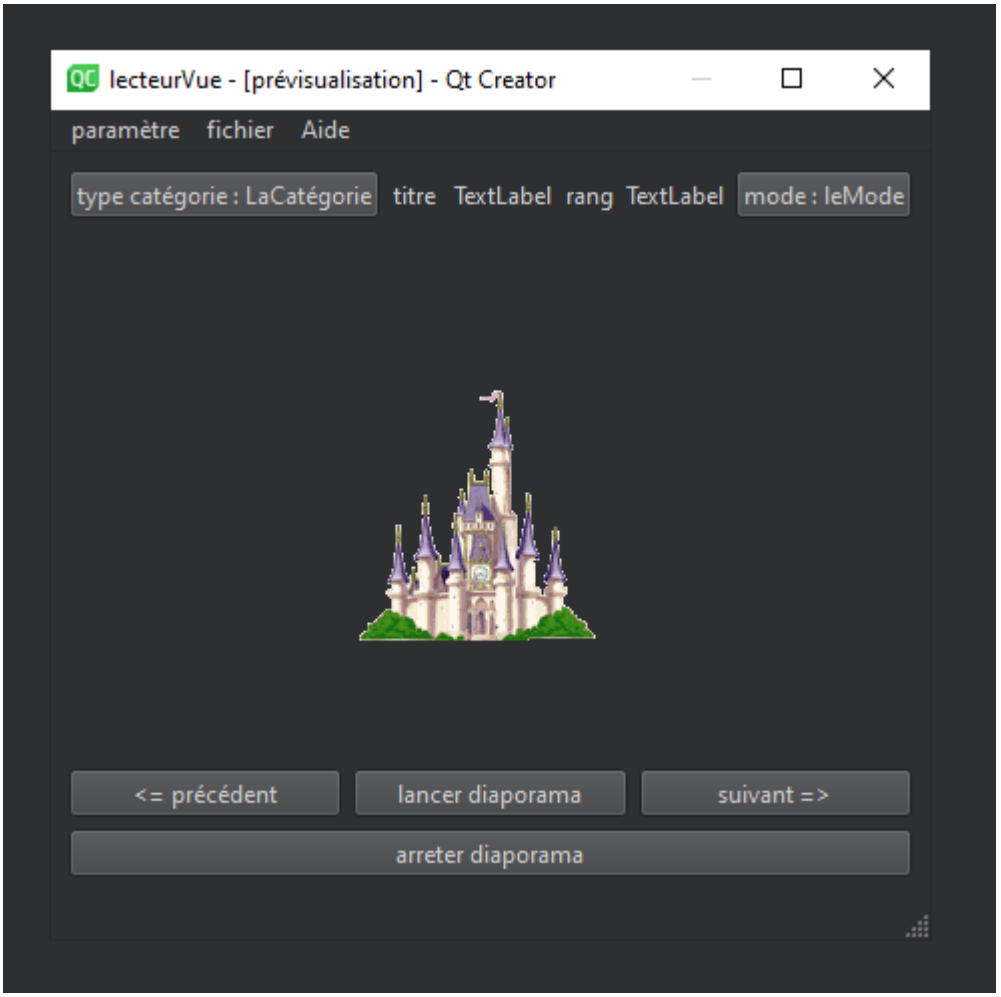
```h
#ifndef VIT_H
#define VIT_H

#include <QDialog>

namespace Ui {
class vit;
}

class vit : public QDialog
{
    Q_OBJECT

public:
    explicit vit(QWidget *parent = nullptr);
    ~vit();
    void setSpine(int);
    int valueSpine();

private:
    Ui::vit *ui;
};

#endif // VIT_H
```

interface graphique

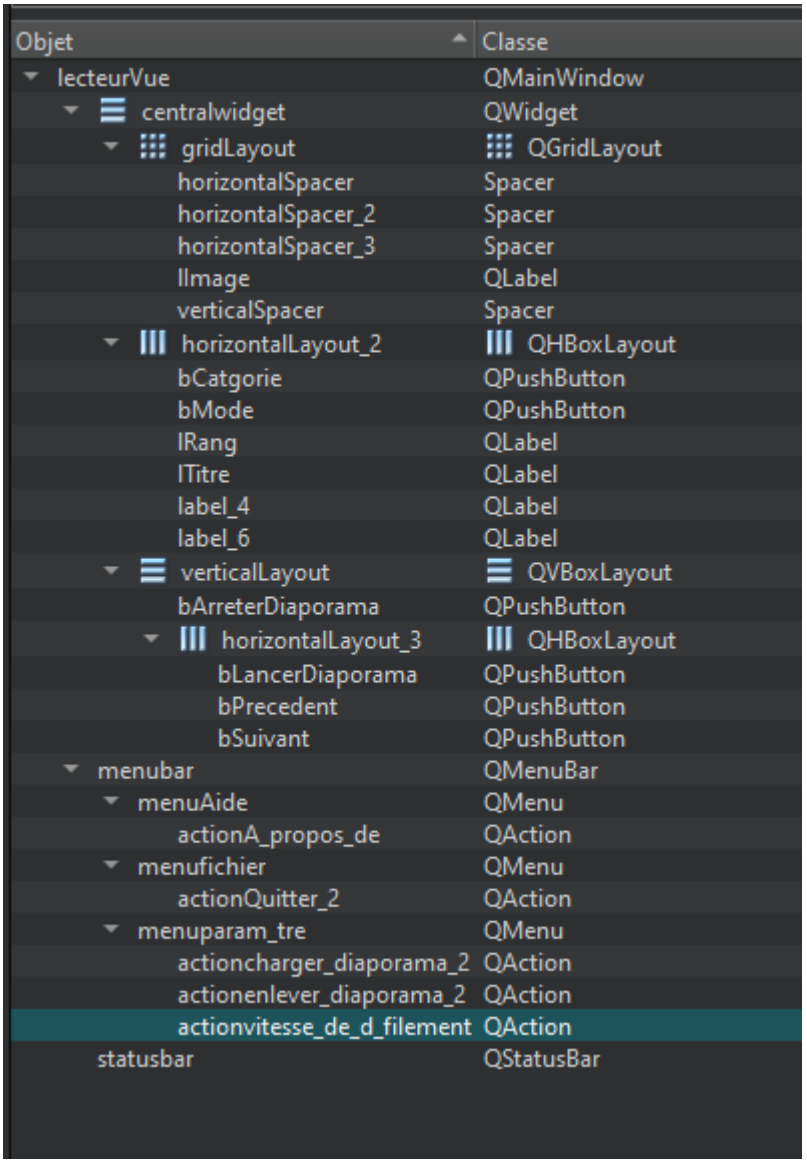
model QT



Prévisualisation



élément de l’interface graphique

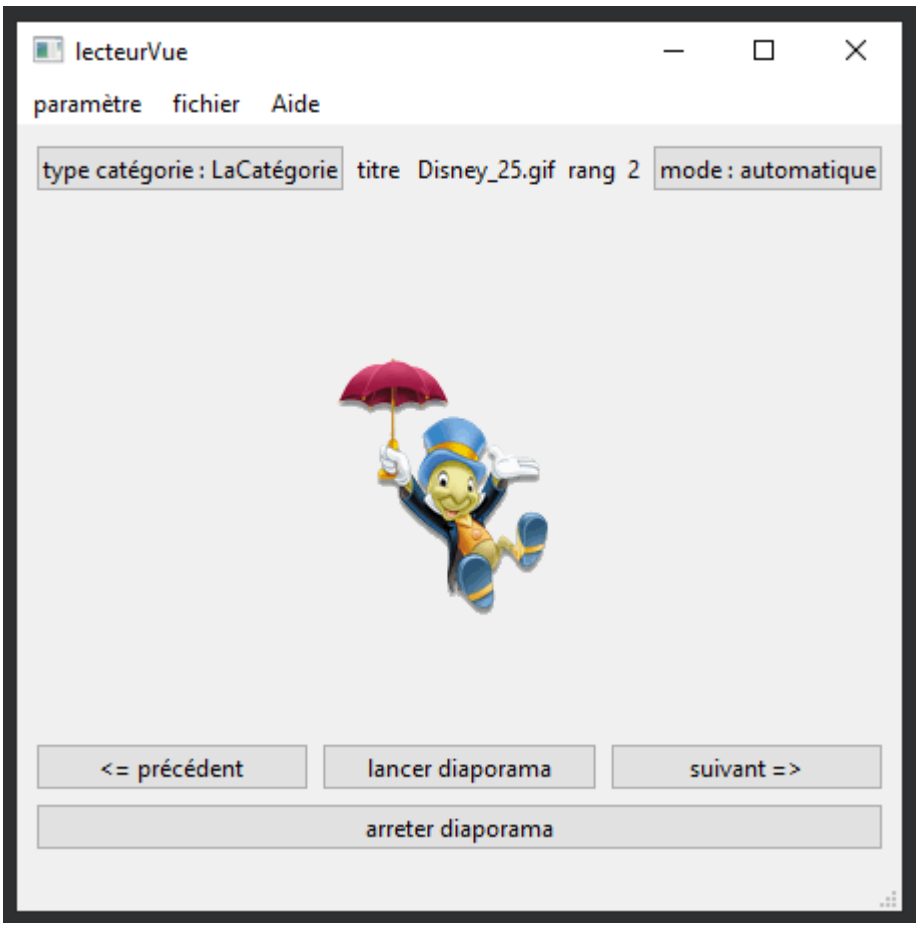


teste

main

```cpp
#include "lecteurvue.h"
#include <iostream>
#include "lecteurvue.h"
#include "image.h"

using namespace std;

#include <QApplication>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    lecteurVue monLecteur;
    monLecteur.afficher();


    monLecteur.show();

    return a.exec();
}
```

résultats attendu


lecteurVue
paramètre
fichier
Aide
type catégorie : LaCatégorie
titre Disney_25.gif rang 2
mode : automatique
<= précédent
lancer diaporama
suivant =>
arreter diaporama

résultat au finale

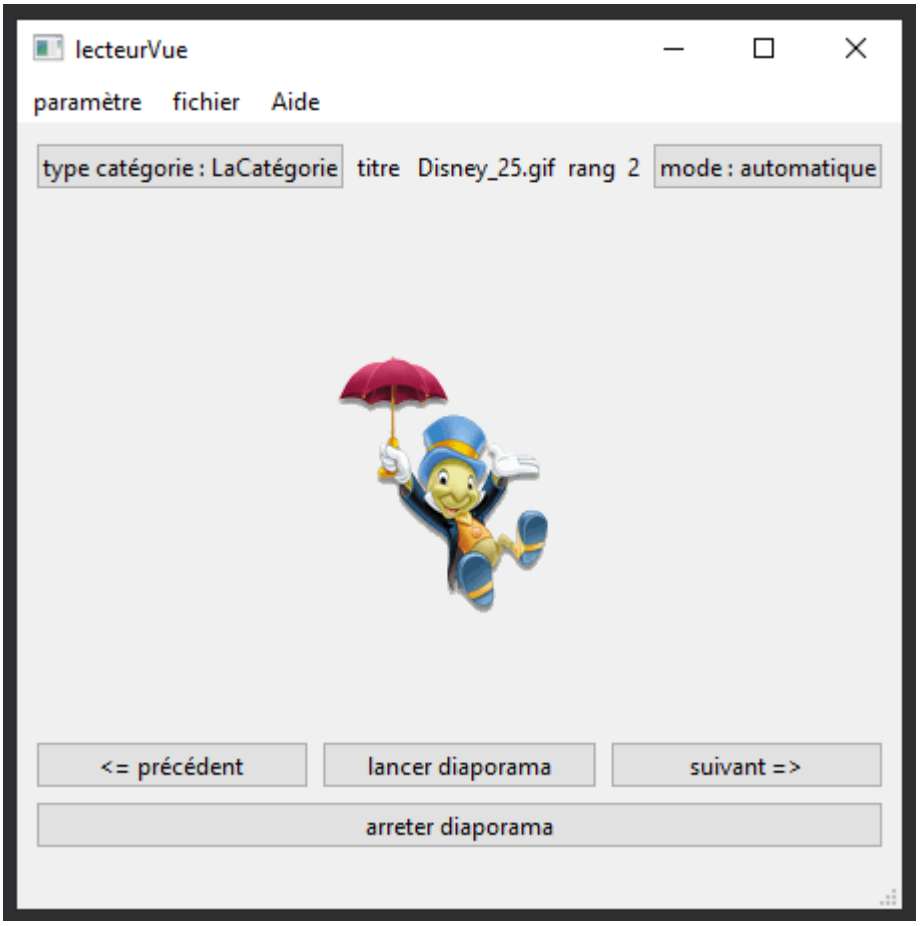
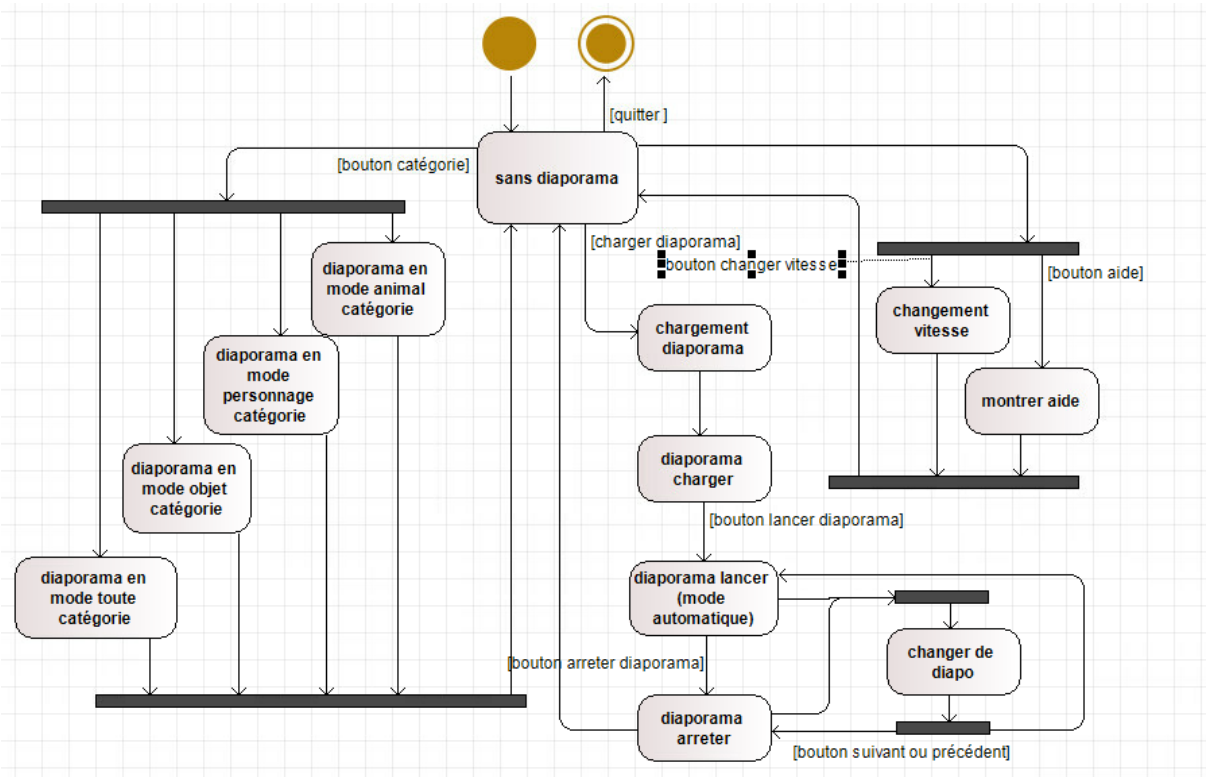


Diagramme états-transitions-actions

le diagramme



Dictionnaire des états, événements et Actions

Dictionnaire des états du diaporama

nomEtat	*Signification*
`sans diaporama`	l'etat qu'est le lecteur quand il est lancer
`diaporama en mode animal catégorie`	l'état de la catégorie du diaporama quand il est en mode animal only
`diaporama en mode personnage catégorie`	l'état de la catégorie du diaporama quand il est en mode personnage only

`diaporama en mode objet catégorie`	l'état de la catégorie du diaporama quand il est en mode objet mal only
`diaporama en mode tous catégorie`	l'état de la catégorie du diaporama quand il est en mode tous
`chargement vitesse`	l'etat ou le lecteur demande à l'utilisateur la vitesse qu'il veut
`montrer aide`	l'etat ou le lecteur lui montre ses informations
`chargement diaporama`	l'etat ou le lecteur demande quelle diaporama il faut charger
`diaporama charger`	l'etat ou le lecteur à un diaporama charger
`diaporama lancer`	l'etat ou le diaporama est lancer en mode automatique
`diaporama arrêter`	l'etat ou le diaporama est en mode manuel
`changer de diapo`	l''etat ou le diaporama va changer de diapositive en mode manuel ou automatique

Dictionnaire des événements faisant changer le diaporama d'état

nomEvénement	*Signification*
`bouton catégorie`	si le bouton catégorie est pressé

`quitter`	si le bouton quitter est pressé
`bouton aide`	si le bouton aide est pressé
`bouton changer vitesse`	si le bouton changer vitesse est pressé
`charger diaporama`	si le bouton charger diaporama est pressé
`bouton lancer diaporama`	si le bouton lancer diaporama est pressé
`bouton arrêté diaporama`	si le bouton arrêter diaporama est pressé
`bouton suivant ou précédent`	si l'un des 2 bouton est pressé

Bilan

Dépôt Git où trouver le projet complet

https://github.com/timouchee/SAE-S.01.git

Temps global de travail

environs 1 heure pour création de cette version en partant de la précédente

Apprentissages majeurs

convertir les type

Difficultés majeures

rien

Points positifs / négatifs de l’activité

pour cette version rien de négatif ni de négatif ça c'est fait tranquillement