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

Département Informatique SAÉ 2.01

SAÉ 2.01 – Développement d’une application

Lecteur de diaporamas – Dossier d’Analyse et conception

## Compléments de spécifications externes.

Dans la V4, la vitesse de défilement maximum n’est pas renseignée (Nous avons décidé un maximum de 30 secondes).

Dans la V6, le nombre de colonnes / informations à afficher dans « Charger images » n’est pas clair (Nous avons décidé d’afficher que le nom des diaporamas).

## Scénarios

**Description scenario nominal:**

Lorsqu’on lance l’application, un lecteur de diaporama s’ouvre. Le diaporama affiche les images du diaporama en cours. Chaque image est affichée accompagnée de son intitule, la catégorie à laquelle elle appartient, et de son rang au sein du diaporama.

Il y a deux modes de fonctionnement ; Un mode manuel où l’image courante reste affichée tant que l’utilisateur ne fait rien, deux boutons “suivant” et “précèdent” permettent de passer manuellement à l’image suivante ou précédente. Un mode automatique où les images défilent automatiquement, ce défilement automatique ne va que vers l’avant et démarre toujours à partir de la première image.

L’utilisateur à la possibilité de passer du mode manuel en mode automatique et inversement à n’importe quel moment.

Au démarrage de l’application, le diaporama est toujours en mode manuel.

**Description scénario alternatif :**

Un menu “Paramètres” est affiché en haut de l’application, il offre plusieurs autres options comme “Vitesse de défilement” qui permet de choisir le temps d’affichage de chaque image, “Charger diaporama” permet de sélectionner le diaporama souhaité et “Enlever diaporama” permet de supprimer du lecteur le diaporama.

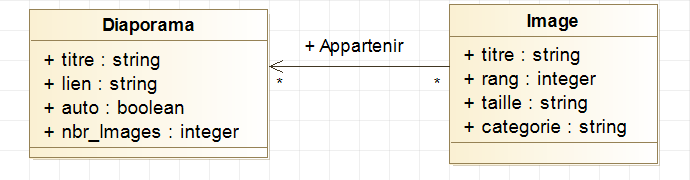
Le menu “Fichier”, avec son option “Quitter”, permet d’arrêter l’application.

Pour finir, le menu “Aide”, avec l’option “A propos de…”, ouvre une boîte de message indiquant les auteurs de et la version de l’application.

## Diagramme de classe (UML)

1. Le diagramme de classes UML

Figure 1 : Diagramme de classes v0



1. Dictionnaire des éléments pour chaque classe

Tableau 1 : Dictionnaire des éléments - Classe DIAPORAMA

|  |  |  |  |
| --- | --- | --- | --- |
| **Classe DIAPORAMA** | | | |
| **Nom attribut** | **Signification** | **Type** | **Exemple** |
| titre | Le titre du diaporama. | string | “mes vacances 2021” |
| lien | Le lien vers le dossier/diaporama contenant les images. | string | “C:/dossier\_diapo” |
| auto | Indique si le diaporama est en mode automatique ou non (manuel). | bool | true |
| nbr\_Images | Le nombre d’images dans le diaporama. | unsigned int | 30 |

Tableau 2 : Dictionnaire des éléments - Classe IMAGE

|  |  |  |  |
| --- | --- | --- | --- |
| **Classe IMAGE** | | | |
| **Nom attribut** | **Signification** | **Type** | **Exemple** |
| titre | Le titre de l’image. | string | “Disney\_0.gif” |
| rang | Le rang de l’image au sein du diaporama. | unsigned int | 1 |
| taille | La taille de l’image. | string | “128x128” |
| categorie | La catégorie à laquelle appartient l’image. | string | « Personnage » |

1. Dictionnaire des méthodes : vous pouvez fournir directement le fichier entête de chaque classe.

Classe lecteur :

Figure 2 : Schéma de classes = Classe lecteur.h

#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

Classe image :

Figure 3 : Schéma de classes = Classe 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();

void afficher(); // affiche tous les champs de l'image

private:

unsigned int \_rang; /\* rang de l'image au sein du diaporama

auquel l'image est associée \*/

string \_titre; // intitulé de l'image

string \_categorie; // catégorie de l'image (personne, animal, objet)

string \_chemin; // chemin complet vers le dossier où se trouve l'image

};

#endif // IMAGE\_H

1. 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

## Implémentation et tests

## 4.1 Implémentation

### Liste et rôle des fichiers de cette version :

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

## 4.2 Test

Test avec le programme fourni main.cpp

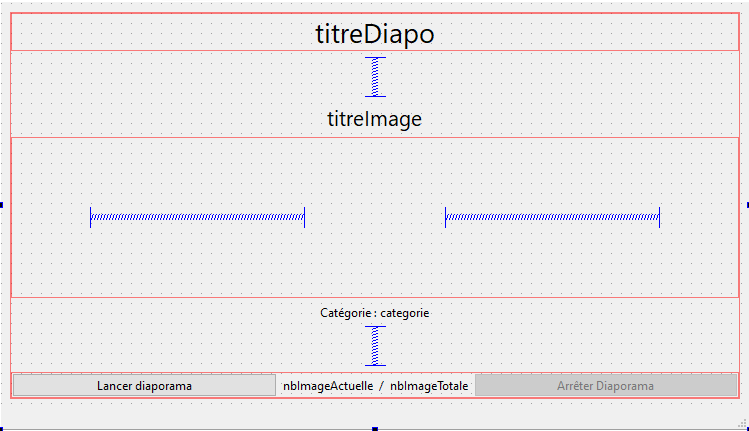
*Valeurs fournies / attendues… comme montré dans la ressource R2.03 (partie tests)*

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Le lecteur charge 4 images. | monLecteur.afficher(); | Affichage du lecteur de diapo. |
| Valide 2 | On change de diaporama courant. | monLecteur.changerDiaporama(1); | Affichage d’un autre diaporama |
| Valide 3 | On avance 4 fois, pour vérifier que avancer() fonctionne correctement | avancer() | L’image courante change 4 fois vers la prochaine image |
| Valide 4 | On recule 5 fois, pour vérifier que reculer() fonctionne correctement | reculer() | L’image courante change 5 fois vers l’image d’avant |
| Valide 5 | On enlève le diaporama du lecteur | monLecteur.changerDiaporama(0); | Le diaporama actuel s’enlève du lecteur |

# Version v1 – projet Graphique seul

## Éléments d’interface

Interface complète :



Pour le titre du diaporama nous avons mis un QLabel « lTitreDiapo » tout en haut avec un layout horizontal.

Au milieu de l’écran nous avons placé un deuxième layout horizontal pour y intégrer les images du diaporama, les boutons QPushButton « bPrecedent » et « bSuivant ». Deux spacer, à gauche et à droite, permettent d’écarter les boutons de l’image qui est située au centre.

En bas de l’écran nous avons décidé de placer un troisème layout horizontal pour y intégrer les QPushButton « bArreterDiapo » et « bLancerDiapo » ainsi que deux QLabel « lImageActuel » et « lImageMax » séparé par une barre oblique.

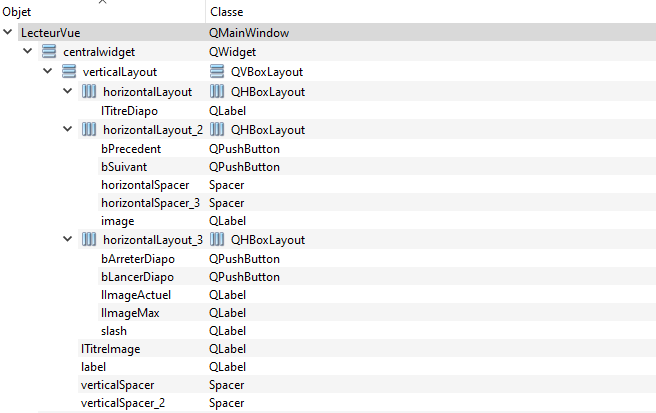
Entre le premier et deuxième layout, nous avons mis le QLabel « lTitreImage » pour avoir le titre de l’image au-dessus de l’image actuel. Un spacer entre le titre de l’image et le titre du diaporama permet que à ces deux QLabel de ne pas être collé.

Entre le deuxième et troisième layout, il y a le QLabel de la catégorie de l’image séparé lui aussi de

« lImageActuel » et « lImageMax » par un spacer pour qu’ils ne soit pas collés.

Dans la barre du menu, nous avons intégré trois autres menus :

* Un menu Aide où il y a un QAction « actionPropos ».
* Un menu Fichier où le QAction « actionQuitter » permet de quitter le diaporama.
* Un menu Parametre où « actionCharger » permet de charger un nouveau diaporama, « actionVitesse » permet de changer la vitesse de défilement.





## 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 :

### 

### Pour les bPushButton (bSuivant, bPrecedent, bLancerDiapo, bArreterDiapo) nous avons mis des signaux cliqués.

### Pour les actions dans les menus (actionCharger, actionVitesse, actionQuitter, actionPropos) nous avons décidé de mettre des signaux déclenchés.

### 6.2 Test

### Comportement de l’interface non lié aux aspects fonctionnels du programme

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Le titre du diaporama est affiché | QLabel : lTitreDiapo | Affichage du titre du diaporama actuel |
| Valide 2 | Le titre de l’image est affiché | QLabel : lTitreImage | Affichage du titre de l’image actuelle |
| Valide 3 | Le bouton pour changer d’image à l’image précédente est affiché | QPushButton : bPrecedent | Affichage du bouton permettant de changer l'image actuelle à l’image précédente |
| Valide 4 | Le bouton pour changer d’image à l’image suivante est affiché | QPushButton : bSuivant | Affichage du bouton permettant de changer l'image actuelle à l’image suivante |
| Valide 5 | L’image est affichée | QLabel : image | Affichage de l’image actuelle |
| Valide 6 | La catégorie de l’image est affichée | QLabel : label | Affichage de la catégorie de l’image actuelle |
| Valide 7 | Le bouton pour lancer le diaporama est affiché | QPushButton : bLancerDiapo | Affichage du bouton pour lancer le défilement automatique |
| Valide 8 | Le bouton pour arrêter le diaporama est affiché | QPushButton : bArreterDiapo | Affichage du bouton pour arrêter le défilement automatique |
| Valide 9 | Un menu Fichier est affiché | QMenu : menuFichier | Affichage d’un menu fichier au-dessus du diaporama |
| Valide 10 | Un menu Paramètre est affiché | QMenu : menuParametre | Affichage d’un menu paramètre au-dessus du diaporama |
| Valide 11 | Un menu Aide est affiché | QMenu : menuAide | Affichage d’un menu aide au-dessus du diaporama |
| Valide 12 | Le numéro de l’image actuel est affiché | QLabel : lImageActuel | Affichage du numéro de l’ismage actuelle en dessous de l’image |
| Valide 13 | Le nombre d’image totale est affiché | QLabel : lImageTotale | Affichage du nombre d’image totale du diaporama. |

# Version v2 –

## Diagramme de classes (UML)

Ajout de la classe « LecteurVue » par rapport à la v0.

## Comportement de l’application

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

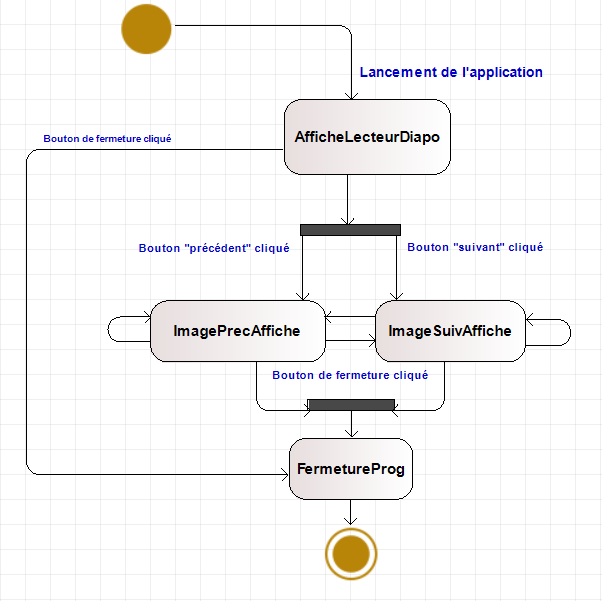


Figure 4 : Diagramme états-transitions du lecteur de diaporamas – v2

### 8.2 Dictionnaire des états, événements et Actions (v2)

**Dictionnaire** **des états du diaporama**

Tableau 2 : États du lecteur de diaporamas – v2

|  |  |
| --- | --- |
| *nomEtat* | *Signification* |
| AfficheLecteurDiapo | Cet état permet d’afficher le lecteur de diaporama |
| ImagePrecAffiche | Cet état permet d’afficher l’image précédente en fonction du rang de l’image actuel |
| ImageSuivAffiche | Cet état permet d’afficher l’image suivante en fonction du rang de l’image actuel |
| FermetureProg | Cet état permet de fermer le programme donc de quitter l’application |

**Dictionnaire des événements faisant changer le diaporama d’état**

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

|  |  |
| --- | --- |
| *nomEvénement* | *Signification* |
| Lancement de l’application | Cet état est activé quand l’utilisateur lance le programme |
| Bouton “Precedent” cliqué | Cet état est activé quand l’utilisateur clique sur le bouton « Précédente » pour afficher l’image précédente |
| Bouton “Suivant” cliqué | Cet état est activé quand l’utilisateur clique sur le bouton « Suivant » pour afficher l’image suivante |
| Bouton de fermeture cliqué | Cet état est activé quand l’utilisateur clique sur le bouton de fermeture pour arrêter le programme |

**Description des actions réalisées lors de la traversée des transitions**

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

Il n’y a aucune action dans notre état-transition

|  |  |
| --- | --- |
| *nomAction* | *Signification* |
| X | X | |
| X | X | |

### 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 🡪* |  |  |  |  |
| *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.*

## Implémentation et tests

### 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 |
| lecteurVue.cpp | Corps de la classe LecteurVue. |
| lecteurvue.ui | Fichier du dessin de l’interface réalisé par QtDesigner |
| image.h | Spécification de la classe Image |
| image.cpp | Corps de la classe Image |
| 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

### Tests (v2)

### 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

### Comportement de l’interface non lié aux aspects fonctionnels du programme

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Le titre du diaporama est affiché | QLabel : lTitreDiapo | Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia  Description générée automatiquement |
| Valide 2 | Le titre de l’image est affiché | QLabel : lTitreImage |
| Valide 3 | Le bouton pour changer d’image à l’image précédente est affiché | QPushButton : bPrecedent |
| Valide 4 | Le bouton pour changer d’image à l’image suivante est affiché | QPushButton : bSuivant |
| Valide 5 | L’image est affichée | QLabel : image |
| Valide 6 | La catégorie de l’image est affichée | QLabel : label |
| Valide 7 | Le bouton pour lancer le diaporama est affiché | QPushButton : bLancerDiapo |
| Valide 8 | Le bouton pour arrêter le diaporama est affiché | QPushButton : bArreterDiapo |
| Valide 9 | Un menu Fichier est affiché | QMenu : menuFichier |
| Valide 10 | Un menu Paramètre est affiché | QMenu : menuParametre |
| Valide 11 | Un menu Aide est affiché | QMenu : menuAide |
| Valide 12 | Le numéro de l’image actuel est affiché | QLabel : lImageActuel |
| Valide 13 | Le nombre d’image totale est affiché | QLabel : lImageTotale |

### Comportement de l’interface lié aux aspects fonctionnels du programme

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Un menu A Propos est affiché quand le menu Aide est déclenché | QAction : actionPropos | Affichage du menu « à propos » dans le menu Aide |
| Valide 2 | Un menu Quitter est affiché quand le menu Fichier est déclenché | QAction : actionQuitter | Une image contenant texte, logiciel, Icône d’ordinateur, Page web  Description générée automatiquement |

### Comportement fonctionnel de l’application

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Une boite de message s’affiche quand on clique sur Aide >> A propos de… (avec la version de l’application, la date de création, les auteurs) | QAction : actionPropos | Une image contenant texte, capture d’écran, logiciel, Page web  Description générée automatiquement |
| Valide 2 | Quitte l’application quand on clique sur Fichier >> Quitter | QAction : actionQuitter | L’application se ferme |
| Valide 3 | Une image en dur s’affiche | image.png | Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia  Description générée automatiquement |
| Valide 4 | L’image passe à la suivante quand on clique sur la flèche à droite de l’image |  | Passage à l’image suivante  Si image 1 alors l’image suivante sera image 2 |
| Valide 5 | L’image passe à la précédente quand on clique sur la flèche à gauche de l’image |  | Passage à l’image précédente  Si image 4 alors l’image précédente sera image 3 |
| Valide 6 | On veut que le diaporama boucle est donc que de la première image on arrive à la dernière et que de la dernière image on arrive à la première |  | Les images du diaporama boucle  Si nombre d’images = 4  Si image 4 alors l’image suivante sera image 1  Si image 1 alors l’image précédente sera image 4 |

Version v5 –

## Diagramme de classes (UML)

Ajout des classes « Database » et « Vitesse » par rapport à la v2.

## Comportement de l’application

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

Une image contenant texte, diagramme, capture d’écran, ligne

Description générée automatiquement

Figure 5 : Diagramme états-transitions du lecteur de diaporamas – v5

### Dictionnaire des états, événements et Actions (v5)

**Dictionnaire** **des états du diaporama**

|  |  |
| --- | --- |
| *nomEtat* | *Signification* |
| AfficheLecteurDiapo | Cet état permet d’afficher le lecteur de diaporama |
| AfficheDiapoActuel | Cet état permet d’afficher le diaporama avec les images actuel |
| ImagePrecAffiche | Cet état permet d’afficher l’image précédente en fonction du rang de l’image actuel |
| ImageSuivAffiche | Cet état permet d’afficher l’image suivante en fonction du rang de l’image actuel |
| FermetureProg | Cet état permet de fermer le programme donc de quitter l’application |
| OuvertureMenuFichier | Cet état permet d’ouvrir le menu “Fichier” |
| QuitterApplication | Cet état permet de quitter l’application |
| OuvertureMenuAide | Cet état permet d’ouvrir le menu « Aide » |
| OuvertureAProposDe | Cet état permet d’ouvrir le fichier « A propos de » |
| OuvertureMenuParametre | Cet état permet d’ouvrir le menu « Parametre » |
| ChangementDiapo | Cet état permet de changer de diaporama |
| EnleverDiapo | Cet état permet d’enlever le diaporama actuel |
| OuvertureMenuVitesse | Cet état permet d’ouvrir le menu « Vitesse » |
| ChangementVitesseDefilement | Cet état permet de changer la vitesse de défilement des images |

Tableau 2 : États du lecteur de diaporamas – v5

**Dictionnaire des événements faisant changer le diaporama d’état**

|  |  |
| --- | --- |
| *nomEvénement* | *Signification* |
| Connexion à la base de données | Cet état est activé quand l’utilisateur lance le programme et se connecte à la base de données |
| Bouton “Precedent” cliqué | Cet état est activé quand l’utilisateur clique sur le bouton « Précédente » pour afficher l’image précédente |
| Bouton “Suivant” cliqué | Cet état est activé quand l’utilisateur clique sur le bouton « Suivant » pour afficher l’image suivante |
| Bouton de fermeture cliqué | Cet état est activé quand l’utilisateur clique sur le bouton de fermeture pour arrêter le programme |
| Menu “Fichier” cliqué | Cet état est activé quand l’utilisateur clique sur le menu “Fichier” |
| Bouton “Quitter” cliqué | Cet état est activé quand l’utilisateur clique sur le bouton « Quitter » |
| Menu “Aide” cliqué | Cet état est activé quand l’utilisateur clique sur le menu « Aide » |
| Fichier “A propos de” cliqué | Cet état est activé quand l’utilisateur clique sur le fichier « A propos de » |
| Diaporama chargé | Cet état est activé quand l’utilisateur charge un diaporama |
| Menu “Paramètre” cliqué | Cet état est activé quand l’utilisateur clique sur le menu « Paramètre » |
| Menu “Vitesse” cliqué | Cet état est activé quand l’utilisateur clique sur le menu « Vitesse » |
| L’utilisateur a change la vitesse de défilement | Cet état est activé quand l’utilisateur change la vitesse de défilement des images |
| Action d’enlever le diaporama actuel déclenché | Cet état est activé quand l’utilisateur déclenche l’action pour enlever le diaporama actuel |
| Action de changer de diaporama déclenché | Cet état est activé quand l’utilisateur déclenche l’action pour changer de diaporama |
| Bouton de fermeture cliqué | Cet état est activé quand l’utilisateur clique sur le bouton pour fermer l’application |

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

**Description des actions réalisées lors de la traversée des transitions**

Il n’y a aucune action dans notre état-transition

|  |  |
| --- | --- |
| *nomAction* | *Signification* |
| X | X | |
| X | X | |

Tableau 4 : Actions à réaliser lors des changements d’état – lecteur de diaporamas v5

### Table T\_EtatsEvenementsActions (v5)

**Correspondance** matricielle du diagramme états-transitions de l’application :

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

|  |  |  |  |
| --- | --- | --- | --- |
| *Élément graphique prenant en charge cet événement 🡪* |  |  |  |
| *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.*

## Implémentation et tests

### 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 |
| 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 |
| lecteur.cpp | Corps de la classe Lecteur |
| image.h | Spécification de la classe Image |
| image.cpp | Corps de la classe Image |
| database.h | Spécification de la classe Database |
| database.cpp | Corps de la classe Database |
| vitesse.h | Spécification de la classe Vitesse |
| vitesse.cpp | Corps de la classe Vitesse |
| vitesse.ui | Fichier du dessin de l’interface du menu de changement de vitesse |
| 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

### 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

### Comportement de l’interface non lié aux aspects fonctionnels du programme

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Le titre du diaporama est affiché | QLabel : lTitreDiapo | Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia  Description générée automatiquement |
| Valide 2 | Le titre de l’image est affiché | QLabel : lTitreImage |
| Valide 3 | Le bouton pour changer d’image à l’image précédente est affiché | QPushButton : bPrecedent |
| Valide 4 | Le bouton pour changer d’image à l’image suivante est affiché | QPushButton : bSuivant |
| Valide 5 | L’image est affichée | QLabel : image |
| Valide 6 | La catégorie de l’image est affichée | QLabel : label |
| Valide 7 | Le bouton pour lancer le diaporama est affiché | QPushButton : bLancerDiapo |
| Valide 8 | Le bouton pour arrêter le diaporama est affiché | QPushButton : bArreterDiapo |
| Valide 9 | Un menu Fichier est affiché | QMenu : menuFichier |
| Valide 10 | Un menu Paramètre est affiché | QMenu : menuParametre |
| Valide 11 | Un menu Aide est affiché | QMenu : menuAide |
| Valide 12 | Le numéro de l’image actuel est affiché | QLabel : lImageActuel |
| Valide 13 | Le nombre d’image totale est affiché | QLabel : lImageTotale |

### Comportement de l’interface lié aux aspects fonctionnels du programme

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Un menu A Propos est affiché quand le menu Aide est déclenché | QAction : actionPropos | Affichage du menu « à propos » dans le menu Aide |
| Valide 2 | Un menu Quitter est affiché quand le menu Fichier est déclenché | QAction : actionQuitter | Une image contenant texte, logiciel, Icône d’ordinateur, Page web  Description générée automatiquement |
| Valide 3 | Un menu Charger est affiché quand le menu Parametre est déclenché | QAction : actionCharger | Une image contenant texte, logiciel, Page web, Icône d’ordinateur  Description générée automatiquement |
| Valide 4 | Un menu Vitesse est affiché quand le menu Parametre est déclenché | QAction : actionVitesse |

### Comportement fonctionnel de l’application

|  |  |  |  |
| --- | --- | --- | --- |
| Classe | Description | Valeur(s) en entrée | Résultat(s) attendu(s) |
| Valide 1 | Une boite de message s’affiche quand on clique sur Aide >> A propos de… (avec la version de l’application, la date de création, les auteurs) | QAction : actionPropos | Une image contenant texte, capture d’écran, logiciel, Page web  Description générée automatiquement |
| Valide 2 | Quitte l’application quand on clique sur Fichier >> Quitter | QAction : actionQuitter | L’application se ferme |
| Valide 3 | Charge les informations relatives au Diaporama et aux Images à partir de la base de données quand on clique sur Paramètre >> Charger diaporama |  | Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia  Description générée automatiquement |
| Valide 4 | L’image passe à la suivante quand on clique sur la flèche à droite de l’image |  | Une image contenant texte, logiciel, Page web, Icône d’ordinateur  Description générée automatiquement |
| Valide 5 | L’image passe à la précédente quand on clique sur la flèche à gauche de l’image |  | Une image contenant texte, capture d’écran, logiciel  Description générée automatiquement |
| Valide 6 | On veut que le diaporama boucle et donc que de la première image on arrive à la dernière et que de la dernière image on arrive à la première |  | Les images du diaporama boucle  Si nombre d’images = 20  Si image 20 alors l’image suivante sera image 1  Si image 1 alors l’image précédente sera image 20 |
| Valide 7 | Quand on clique sur « Lancer diaporama » le Statut se met en automatique et les images défilent |  | Une image contenant texte, capture d’écran, logiciel, Page web  Description générée automatiquement |
| Valide 8 | Quand on clique sur « Arrêter diaporama » le Statut se met en manuel et on reste sur l’image qui était affichée au moment du clic |  | Une image contenant texte, logiciel, Page web, Icône d’ordinateur  Description générée automatiquement |
| Valide 9 | Quand on clique sur une des deux flèches le Statut se met en manuel et on reste sur l’image qui était affichée au moment |  | Une image contenant texte, logiciel, Page web, Icône d’ordinateur  Description générée automatiquement |
| Valide 10 | Une boite de dialogue s’affiche quand on clique sur Paramètre >> Vitesse de défilement, cette fenêtre permet de changer la vitesse de défilement du mode automatique (le temps d’attente entre chaque image) |  | Une image contenant texte, logiciel, Icône d’ordinateur, Page web  Description générée automatiquement |
| Valide 11 | Le diaporama se décharge et on revient sur la page de base quand on clique sur Paramètre >> Enlever le diaporama |  | Une image contenant texte, capture d’écran  Description générée automatiquement |
| Invalide 1 | Une erreur de connexion à la base de données quand on clique sur Paramètre >> Charger diaporama |  | Le diaporama ne charge pas et on reste sur la page initiale |

## Bilan

**Dépôt Git où trouver le projet complet (les versions réalisées)**: <https://github.com/maxEtch/S101-Dev#:~:text=/-,S101%2DDev>

**Temps global de travail (pour le groupe) :** 38,5h environ

**Apprentissages majeurs :** Implémentation d’une application graphique et liaison avec une base de données

**Difficultés majeures :** La documentation

**Points positifs / négatifs de l’activité :** Libertés graphiques sur l’application / Documentation très longue