

# Projet tutoré du semestre 2

Yanis Levesque  
Thomas Vathonne

UNIVERSITÉ DE  
VERSAILLES  
ST-QUENTIN-EN-YVELINES



## Table des matières

<b>Projet tutoré du semestre 2</b> .....	1
Introduction.....	3
Cadre d'utilisation .....	3
Conception (générale et détaillée).....	4
-Diagramme de cas d'utilisation.....	4
-Diagramme de classe .....	5
-Diagrammes séquentiels.....	6
Dossier de test.....	8
Test de la classe Frise Chronologique.....	8
Ajout d'un événement.....	8
Suppression d'un événement.....	8
Obtenir le poids d'un événement .....	8
Obtenir la liste des événements.....	8
Suppression des événements hors période .....	8
Sauvegarde de la frise dans un fichier.....	9
Existence d'un événement dans une frise.....	9
Manuel utilisateur .....	10
Pour commencer .....	10
Créer une nouvelle frise chronologique.....	10
Ajouter un évènement à la frise.....	12
Modifier un évènement de la frise.....	14
Supprimer un évènement.....	15
Le diaporama.....	16
Sauvegarder sa frise .....	16
Ouvrir une frise au lancement de l'application .....	17
Menu Aide .....	18
Quitter .....	18
Conclusion .....	19

## Introduction

Voici le rapport de projet tutoré du second semestre réalisé par Yanis Levesque et Thomas Vathonne, vous trouverez ci-contre tous les éléments concernant la réalisation de ce projet qui consiste à créer une application permettant la visualisation et création de frise chronologique personnalisée par l'utilisateur.

Note : tous les diagrammes présents dans ce rapport sont aussi dans un fichier à part sur Github ouvrable avec le logiciel « StarUml » pour plus de clarté et de lisibilité.

## Cadre d'utilisation

L'application doit permettre à l'utilisateur de créer et d'enregistrer des frises chronologiques qu'il réalise lui-même. A son lancement, l'application doit permettre à l'utilisateur de créer une nouvelle frise ou d'ouvrir une déjà faite auparavant. Quel que soit le choix effectué par l'utilisateur, il doit donc pouvoir facilement ajouter, modifier ou supprimer à sa guise des événements liés à la frise à tout moment.

Il est donc indispensable que les frises soient donc enregistrables et ouvrables directement par l'application et par ce fait, les frises doivent être disponible sous le format d'un fichier compréhensible par l'application.

Sous les conseils de nos professeurs et en accord avec le sujet, l'application est séparée en 4 menus internes : « Création », « Affichage », « Aide » et « Quitter ».

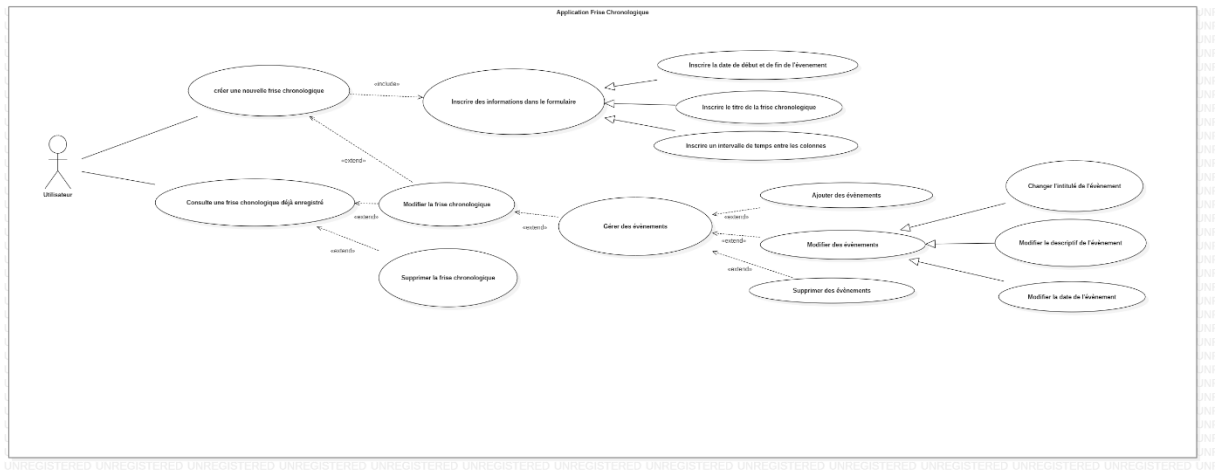
Toutes informations à apporter relatives à la saisie de l'utilisateur sont effectuées à l'aide de formulaires spécialement conçus dans le menu « Création ».

L'affichage du diaporama et de la table d'événements est situé dans le menu « Affichage ».

Le menu « Aide » indiquera un lien vers le manuel utilisateur et le menu « Quitter » permet juste de quitter l'application.

## Conception (générale et détaillée)

### -Diagramme de cas d'utilisation



Voici le diagramme de cas d'utilisation de notre application qui, pour rappel, est aussi donné en annexe pour une meilleure visibilité.

L'utilisateur peut, en premier lieu, créer une frise chronologique ou en modifier une déjà créée.

S'il choisit d'en créer une nouvelle, il sera amené à remplir les informations nécessaires à la création de celle-ci dans un formulaire :

- Le titre
- La date de début et de fin
- Un intervalle de temps (une période) pour un meilleur affichage de la frise.

Après cette première étape, la frise est instanciée, l'utilisateur peut alors choisir de l'initialiser et ainsi remplir sa frise d'événements.

Ces événements sont définis par l'utilisateur en trois caractéristiques : leur intitulé, leur descriptif et par une date. Qu'il modifie ou crée la frise chronologique, l'utilisateur peut ajouter, modifier, supprimer les événements et leurs caractéristiques.

[illegible]

Une classe « LectureEcriture » existe dans l'unique but de pouvoir enregistrer les frises

La classe `ModeleTable` quant à elle pose les bases d'une `ITable` d'évènement qui sera affiché

Pour finir, l'interface « ConstantesTextes » permet de réunir tout les constantes textes dont

Une classe « FenetreMere » a pour contentPane un « PanelFils » qui contiendra tous nos

- Le « PanelCreation » ayant pour layout un border layout qui est lui-même composé de

centre) et « PanelAffichageEvt » (ayant pour layout un gridBagLayout, avec le titre, la date, et la description de l'évènement).

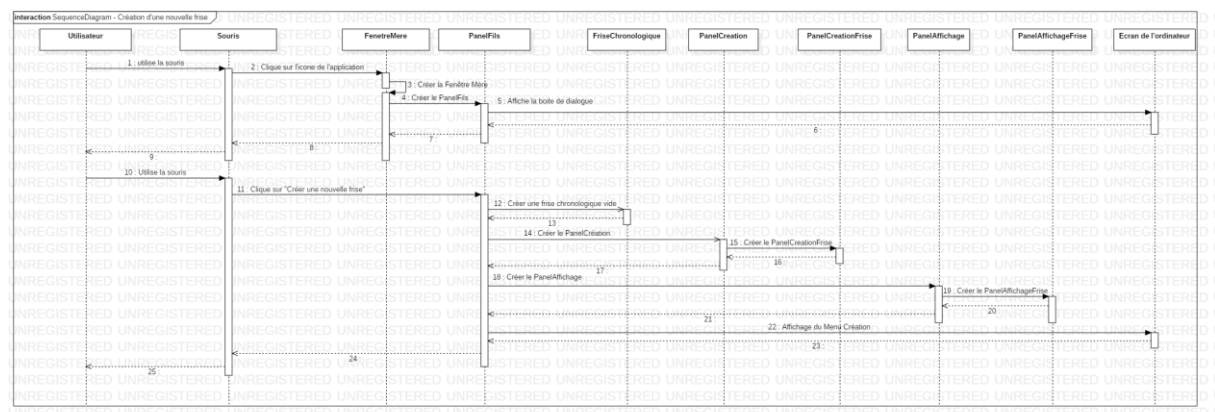
PanelAffichageFrise est placé dans la partie sud du BorderLayout de PanelAffichage, tandis que la partie nord est constituée d'un BorderLayout qui contient un CardLayout et permet ainsi l'affichage du diaporama.

PanelAffichageFrise utilise les services de « CelluleRender » pour un meilleur affichage de la JTable.

Le paquet controleur ne contient qu'une seule classe nommée « Controleur », qui permet d'effectuer la synchronisation entre les données du modèle et la vue présentée à l'utilisateur. IL se met à l'écoute de « PanelAffichage » et de « PanelCreation » et utilise l'interface « ConstantesTextes »

## -Diagrammes séquentiels

### Diagramme séquentiel – Création d'une nouvelle frise

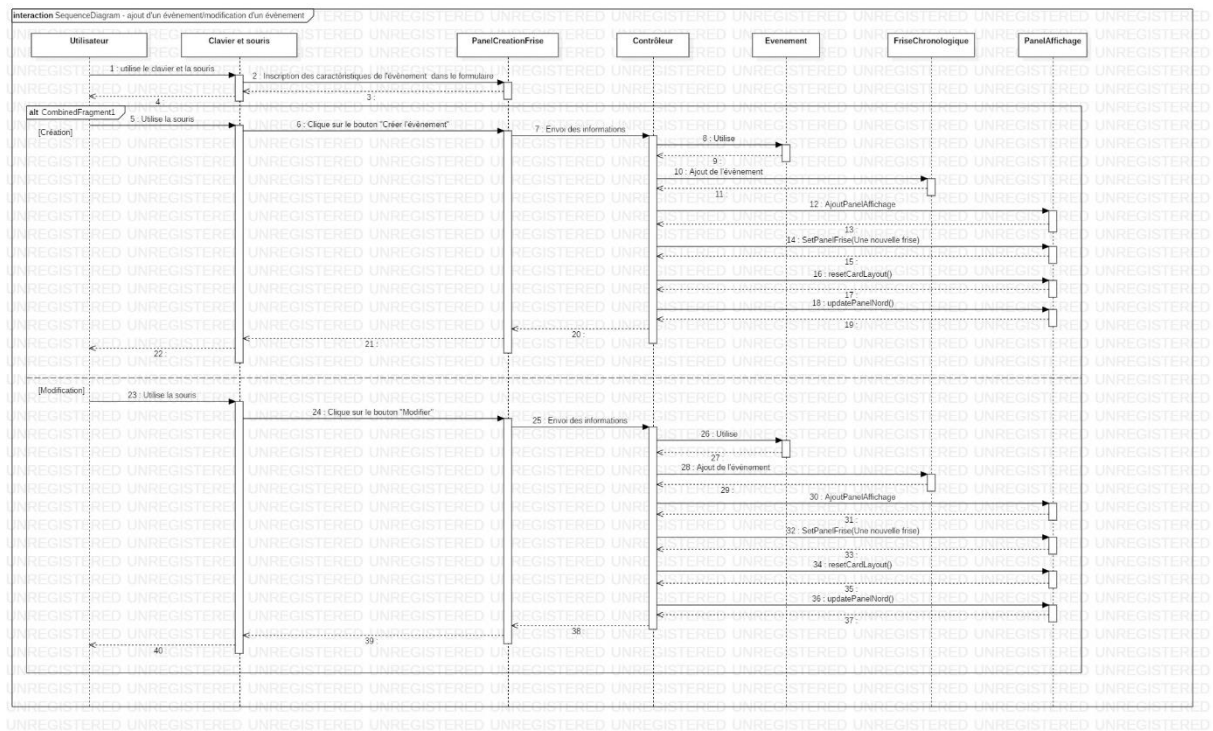


Voici le diagramme séquentiel qui montre tout ce que déclenche la création d'une nouvelle frise.

L'utilisateur utilise sa souris pour cliquer sur l'exécutable de notre application, cette action entrainera en chaîne la création de la « FenetreMere », du « PanelFils » et affichera une boîte de dialogue lui demandant s'il veut ouvrir une frise chronologique déjà existante ou au contraire en créer une nouvelle.

Si l'utilisateur clique sur « créer une nouvelle frise », la frise sera créée ainsi que le « PanelCreation », le « PanelCreationFrise », le « PanelAffichage », le « PanelAffichageFrise ». Le menu création s'affichera ensuite sous les yeux de l'utilisateur

## Diagramme séquentiel – Ajout d'un évènement/Modification d'un évènement



Voici le diagramme séquentiel qui montre tout ce que déclenche la création d'un nouvel évènement et la modification de celui-ci.

Tout commence par le clic de l'utilisateur sur les champs du formulaire et le remplissage de ceux-ci.

La véritable différence entre ces deux actions est que le formulaire à remplir est légèrement différent. En réalité il y a deux formulaires différents, mais il se ressemblent en tout point sauf sur l'inscription du bouton de validation ou sera marqué « Créer l'évènement » ou « Modifier l'évènement » selon le cas.

Dans tout les cas, l'utilisateur va cliquer sur un bouton de validation qui va entraîner l'envoi d'information au Contrôleur qui utilise la classe « Evènement » du modèle pour pouvoir ensuite l'ajouter à la frise.

L'ajout au PanelAffichage est ensuite effectué et trois méthodes de la classe PanelAffichage sont utilisées, setPanelFrise(maNouvelleFrise), resetCardLayout() et updatePanelNord() ce qui permet d'avoir un affichage qui prend en compte la modification ou l'ajout.

## Dossier de test

### Test de la classe Frise Chronologique

La frise chronologique est une des classes centrales du programme, elle doit donc être bien testée pour être sûr de son bon fonctionnement.

#### Ajout d'un événement

Pour commencer on va tester la fonction « ajoutEvenement » de la classe « FriseChronologique ». On crée pour cela un événement, on l'ajoute à une nouvelle frise chronologique puis on regarde si cet événement est présent dans la HashMap des différents événements de la frise grâce à la méthode « containsValue », si l'événement est trouvé alors le test est réussi sinon si l'on a testé tous les événements sans trouver le nouveau alors la fonction n'a pas marché.

#### Suppression d'un événement

Il s'agit d'un test qui marche de la même façon que celui fait précédemment. On ajoute un événement à la frise chronologique, puis on le supprime et on regarde s'il existe encore dans la HashMap des événements de la frise. Si oui alors la fonction n'est pas opérationnelle.

#### Obtenir le poids d'un événement

On teste ici la fonction « getPoidsEvenement » qui à partir d'un événement donné va retrouver son poids dans la frise chronologique et retournera -1 s'il ne le trouve pas. Pour tester cette fonction on ajoute trois événements de différents poids et de différentes dates dans la frise chronologique. On compare ensuite pour chaque événement si le poids renvoyé par la fonction est bien le même que celui attendu.

#### Obtenir la liste des événements

Pour ce test on ajoute tout d'abord trois événements à une frise chronologique et que l'on ajoute ensuite à une « ArrayList ». On compare ensuite si la fonction « getListeEvenements » de la frise chronologique est égale à la liste créée précédemment. Si oui alors la fonction est opérationnelle.

#### Suppression des événements hors période

Cette fonction a pour but de supprimer tous les événements qui ne sont pas entre la date de début et la date de fin de la frise chronologique. Pour tester cela on ajoute trois événements à la frise chronologique et on construit une « ArrayList » qui contiendra le résultat attendu à savoir avoir uniquement les deux premiers événements dans la frise. Après cela on change la date de fin de la frise afin que l'événement numéro trois ne soit plus dans la bonne période puis on appelle la fonction



« supprimerEvenementHorsPeriode ». On test ensuite si la liste créer précédemment est égale à ce que renvoi la méthode « getListeEvenements » testée plus tôt. S’il s’agit de la même chose alors la fonction est opérationnelle.

#### Sauvegarde de la frise dans un fichier

Cette fonction a pour but de sérialiser la frise chronologique dans un fichier. Pour tester cela on ajoute des événements à une frise et on utilise la fonction « sauvegarderFrise » dans une structure « try .. catch ». S’il y a une erreur lors de l’enregistrement et donc que l’on passe dans le bloc « catch » cela signifie que la fonction ne marche pas.

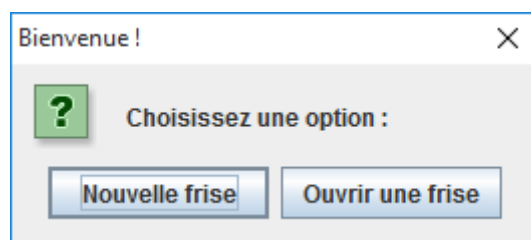
#### Existence d’un événement dans une frise

Pour tester si un événement existe dans une frise la fonction « evenementExisteFrise » a besoin de deux paramètres : un poids et une année. Pour la tester on ajoute trois événements à des poids et des années différentes dans la frise chronologique. On fait ensuite différents tests dans différents cas et on compare au résultat attendu. Si toutes les comparaisons passent alors la fonction possède le comportement attendu.

## Manuel utilisateur

Pour commencer

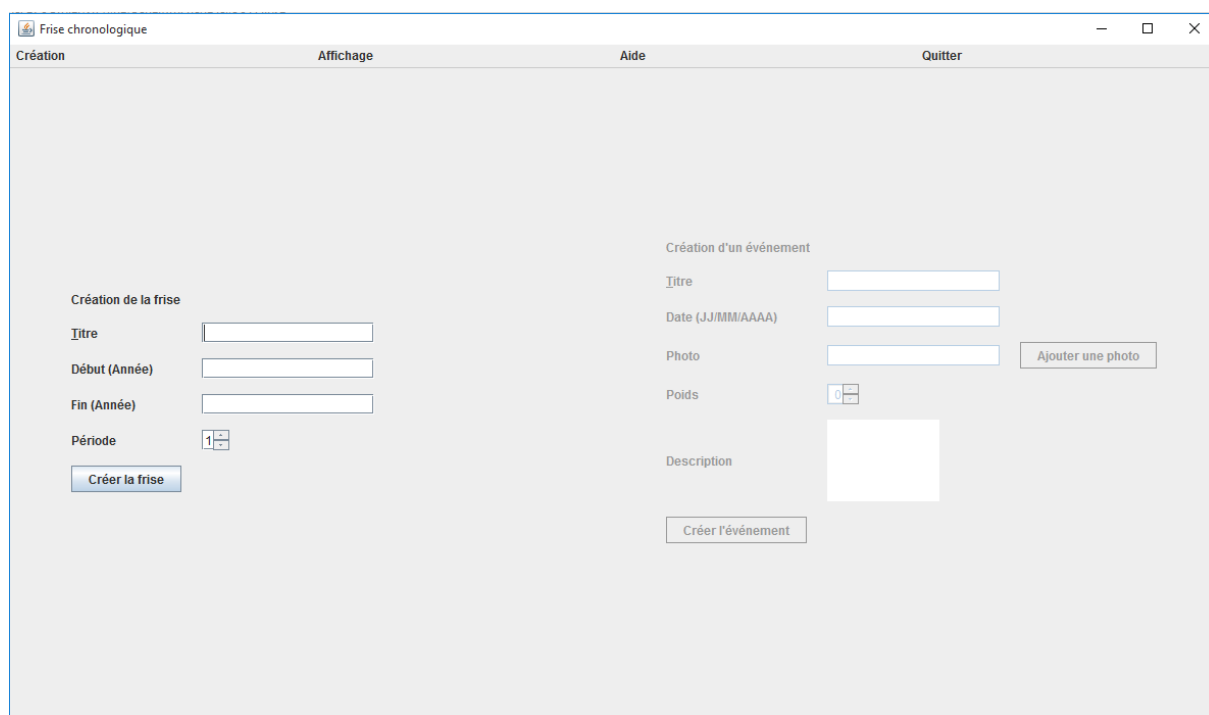
Au lancement de l'application, vous vous retrouverez face à cette boîte de dialogue :



Vous pouvez alors créer une nouvelle frise chronologique en cliquant sur « Nouvelle frise » ou bien, ouvrir une frise déjà existante enregistrée sur votre ordinateur ou autres plateformes connectées (clef USB, disque etc..) en cliquant sur « Ouvrir une frise ».

Créer une nouvelle frise chronologique

Après avoir appuyé sur le bouton « Nouvelle frise », cette fenêtre s'ouvrira :



Vous êtes situé dans le menu « Création », vous pouvez alors créer votre frise en remplissant les différents champs de texte et la période :

1. Le titre : ceci est le titre de votre frise chronologique.
2. Début (année) : c'est l'année de départ de votre frise.
3. Fin (Année) : c'est l'année de fin de votre frise.
4. Période : nombre d'années d'écart choisi pour votre frise (Exemple : afficher ma frise uniquement par intervalle de 5 années ne va afficher les années en entête que tous les 5 ans pour la rendre plus lisible).

A titre informatif, tous ces paramètres sont obligatoires pour la création de la frise.

Ensuite, une fois que tous les paramètres sont remplis comme ceci :

**Création de la frise**

**Titre**

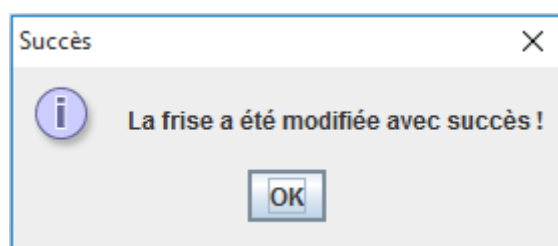
**Début (Année)**

**Fin (Année)**

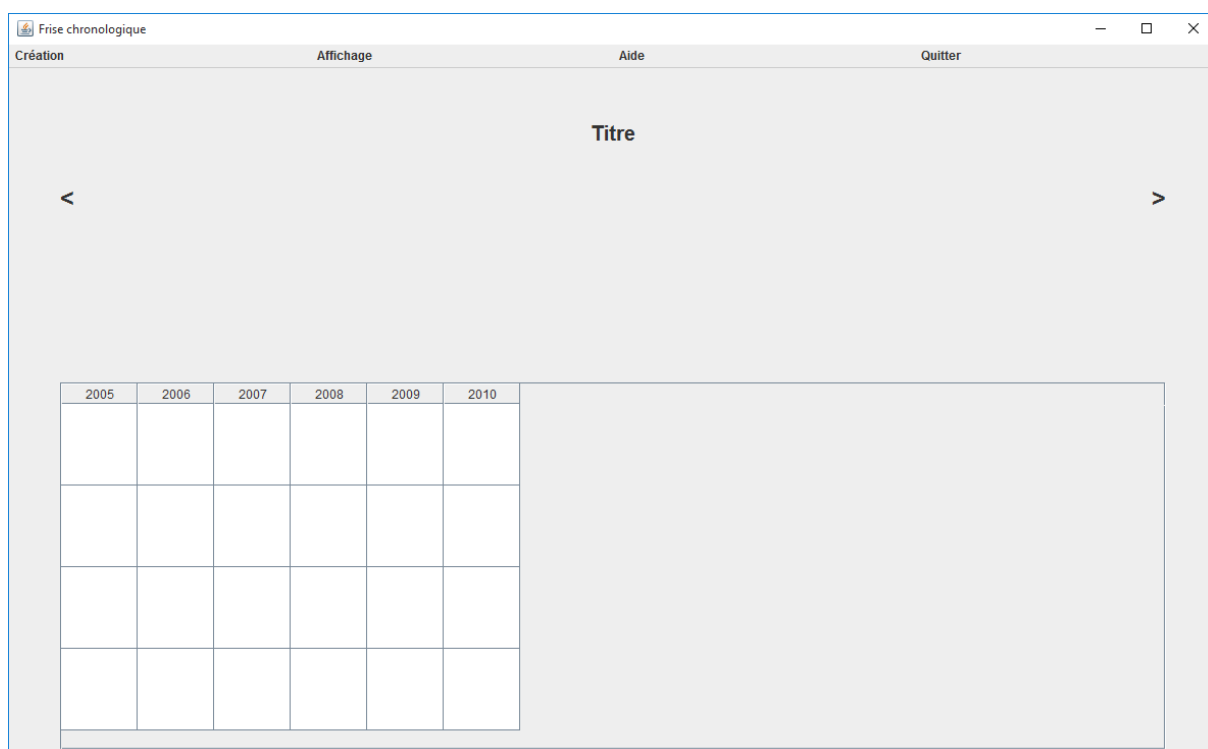
**Période**

**Créer la frise**

Appuyer sur le bouton « Créer la frise », ce message devrait normalement s'afficher :



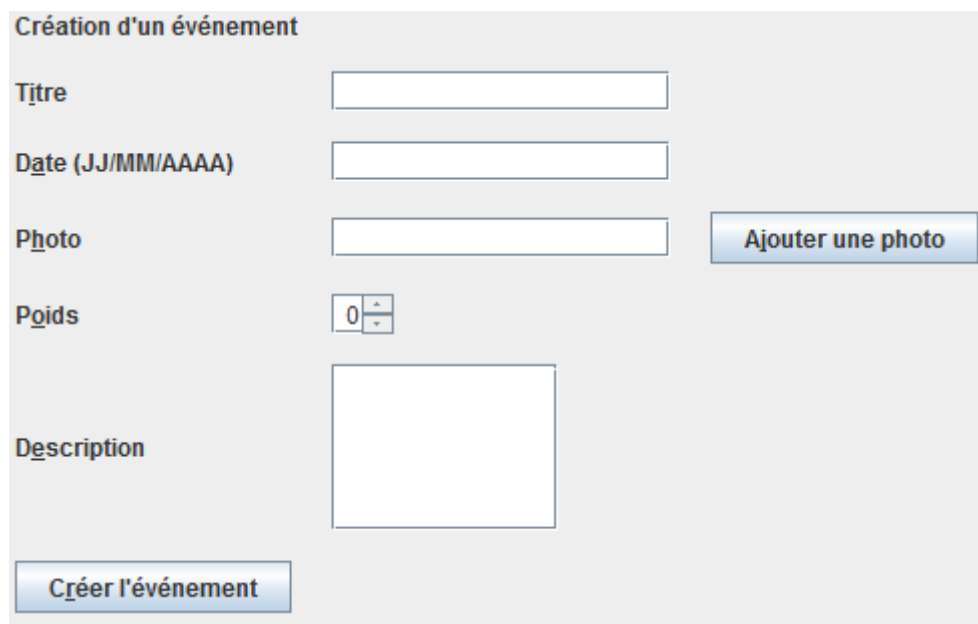
Appuyer ensuite sur le menu Affichage et vous pourrez constater que votre frise a bien été créée :



Ajouter un évènement à la frise

Maintenant il ne reste plus qu'à la remplir d'évènements ! Retournons dans le menu Création situé en haut à droite de votre fenêtre.

Nous allons nous intéresser à l'ajout des évènements grâce au formulaire situé à droite :



Création d'un événement

Titre

Date (JJ/MM/AAAA)

Photo  [Ajouter une photo](#)

Poids

Description

[Créer l'événement](#)

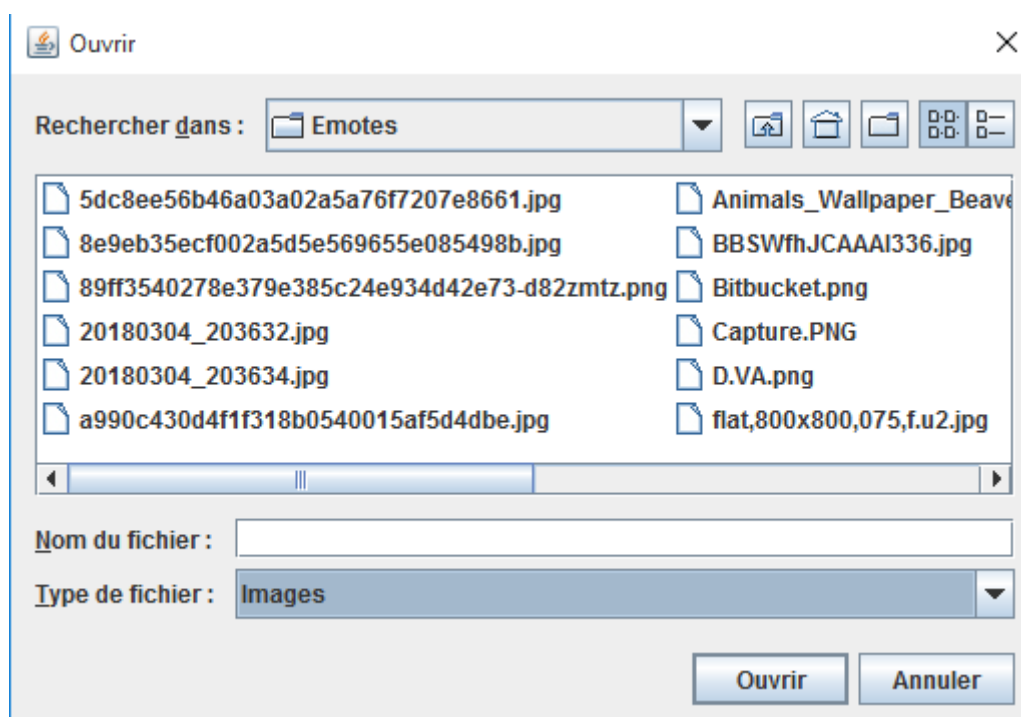
C'est en remplissant ce formulaire qu'un nouvel évènement s'ajoutera à votre frise :

1. Titre : ceci est le titre de l'évènement.
2. Date : la date de l'évènement est à fournir selon le format Jour/Mois/Année.
3. Photo : une photo qui est associée à l'évènement visible en format réduit dans votre frise et en format étendu dans le diaporama si vous cliquez dessus.
4. Poids : c'est l'importance donnée à l'évènement à ajouter par rapport aux autres évènements de la frise (0 étant le plus important et 3 le moins important).
5. Description : l'évènement a une description qui lui est associée.

Voici un exemple de saisie classique :

The screenshot shows a web form titled 'Création d'un événement'. It contains several input fields: 'Titre' with the value 'Naissance d'une loutre', 'Date (JJ/MM/AAAA)' with '01/02/2005', 'Photo' with a long alphanumeric string, and 'Poids' with '0'. There is a text area for 'Description' containing 'Une loutre est née.' and a 'Créer l'événement' button at the bottom. An 'Ajouter une photo' button is located next to the photo input field.

Note : vous pouvez choisir directement la photo que vous voulez utiliser en parcourant l'arborescence de vos documents après avoir appuyé sur « Ajouter une photo » :

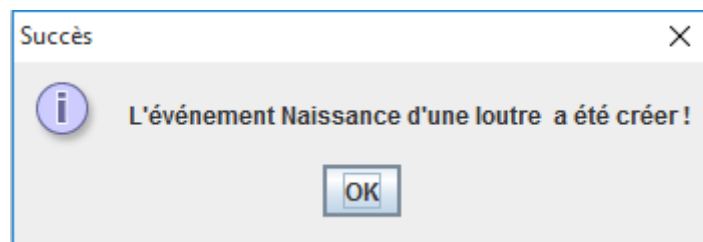


Une fois l'image que vous voulez utiliser est trouvée, cliquer sur le fichier puis sur « Ouvrir ».

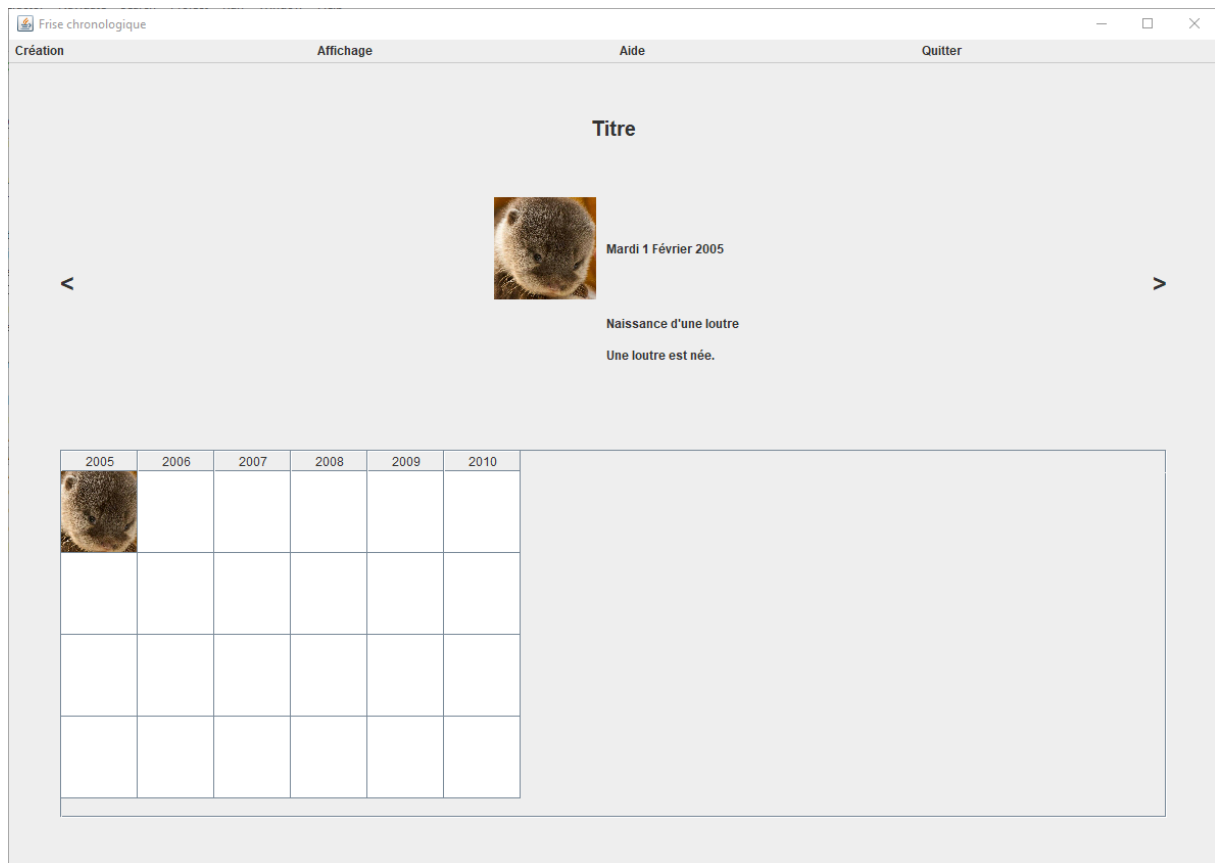
Ou vous pouvez directement indiquer le chemin où l'image est située avec son nom dans le champ prévu à cet effet :

The screenshot shows a single input field labeled 'Photo' with an empty text box next to it.

Une fois fini, appuyer sur « Créer l'évènement », ce message devrait normalement s'afficher :

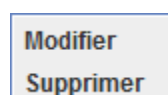


Si vous retournez sur le menu affichage, l'évènement est désormais visible et a été ajouté à votre frise :



## Modifier un évènement de la frise

Si vous voulez modifier un évènement de la frise faite un clic droit sur l'évènement, cette fenêtre devra apparaître :



Appuyer sur « Modifier » et vous serez téléporté sur le menu Création avec l'option modification d'activer :

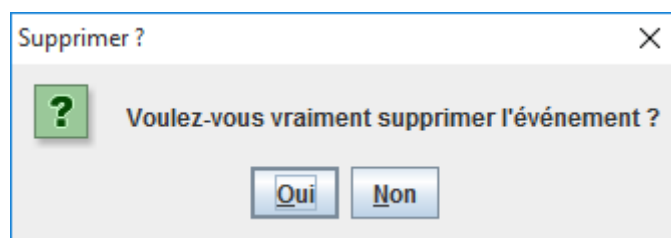
The form is titled "Modification de l'événement". It contains the following fields and controls:

- Titre**: A text input field containing "Naissance d'une loutre".
- Date (JJ/MM/AAAA)**: A text input field containing "01/02/2005".
- Photo**: A text input field containing a file path "02a5d5e569655e085498b.jpg". To its right is a button labeled "Ajouter une photo".
- Poids**: A numeric input field with a spinner, currently showing "0".
- Description**: A text area containing the text "Une loutre est née."
- At the bottom, there are two buttons: "Modifier l'événement" and "Annuler la modification".

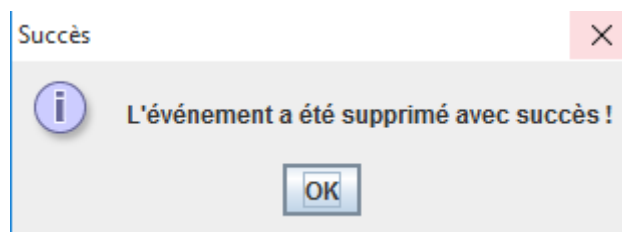
Vous pouvez alors soit appuyer sur « Annuler la modification » si vous ne voulez plus le modifier, ou alors appuyer sur « Modifier l'événement » après avoir modifié les paramètres voulus.

### Supprimer un évènement

Si vous cliquez sur « Supprimer », après avoir réalisé un clic droit sur un évènement, cette fenêtre s'affichera :



Si vous ne voulez pas le supprimer, appuyez sur « Non » pour annuler la suppression. Dans le cas contraire, appuyer sur « Oui ». Cette fenêtre s'affichera alors :



## Le diaporama

Les évènements que vous rajoutez seront automatiquement ajoutés dans le diaporama dans le menu affichage:



Vous pouvez vous déplacer entre les éléments du diaporama grâce à ces deux boutons :



: Permet d'accéder à l'évènement précédent

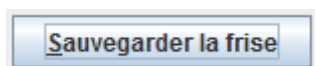


: Permet d'accéder à l'évènement suivant

## Sauvegarder sa frise

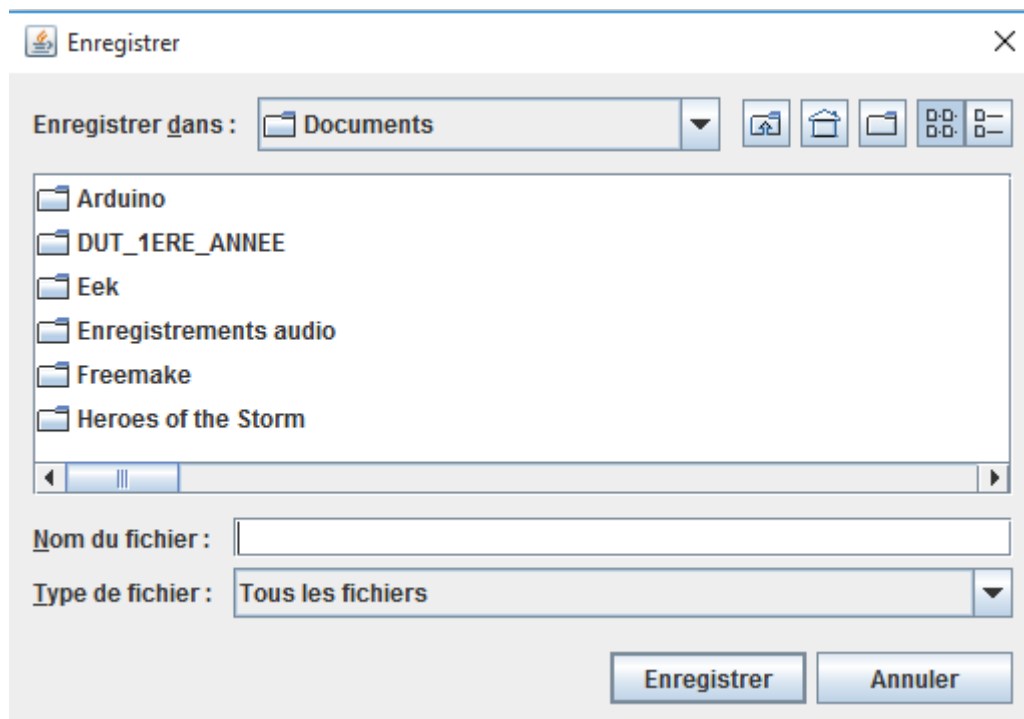
Maintenant que vous avez une frise chronologique remplie d'évènements, vous pouvez la sauvegarder.

Allez dans le menu Création et cliquez en bas à droite sur « sauvegarder la frise » :





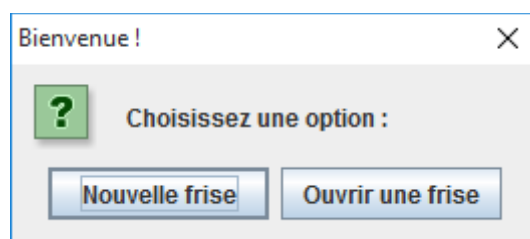
Cette fenêtre apparaîtra par la suite :



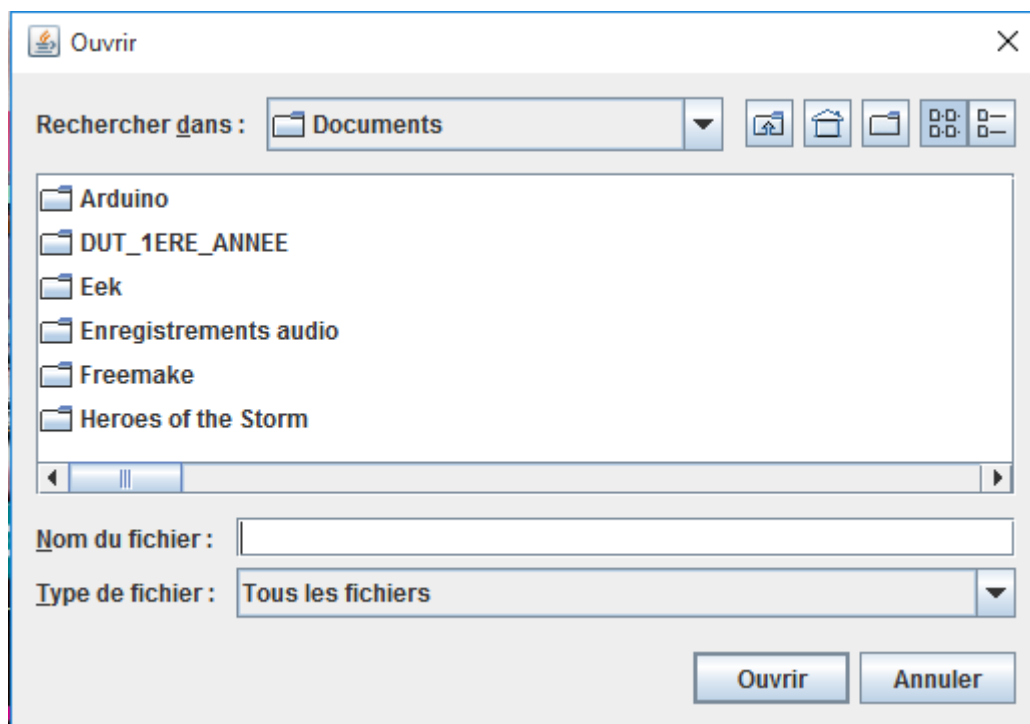
Vous n'avez plus qu'à choisir le nom que vous voulez donner à votre fichier, son emplacement et appuyer sur « enregistrer » !

Ouvrir une frise au lancement de l'application

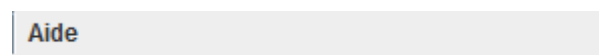
Si au lancement de l'application, vous voulez ouvrir une frise déjà créée, appuyer sur « Ouvrir une frise »



Vous vous retrouvez une nouvelle fois à l'explorateur de fichiers où vous pouvez sélectionner votre frise :

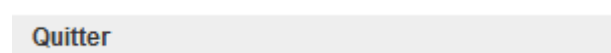


Menu Aide



Permet d'obtenir un lien vers ce manuel utilisateur en cas de problème d'utilisation.

Quitter



Permet de quitter l'application au même titre que la croix de la fenêtre de l'application.

## Conclusion

Lors de ce projet, nous avons essayé de nous répartir les tâches équitablement. Thomas a donc principalement fait le code avec les différentes classes du paquet « vue » et « controleur ». Quant à moi, j'ai réalisé les différents diagrammes UML de conceptualisation, la cohérence du paquet « modèle » avec le reste du programme ainsi que la modification des classes de ce paquet et le rapport dans son intégralité hormis le jeu de test et le dossier qui lui est associé. Nous nous sommes ensuite répartis les commentaires du code et la Javadoc équitablement.

Nous pensons aussi que notre application respecte bien ce qui était demandé cependant certains améliorations sont possibles : tout d'abord l'amélioration de l'aspect graphique général ainsi que l'ajout de la synchronisation entre le diaporama d'événements et la barre de déplacement de la table.