


Frise chronologique

Fichier Ajouter Quitter ?



26/8/1945

Evenement

Description

			1940				1944				1948	
							liberation.jpg					

Projet tutoré : Frise chronologique

Semestre 2 - 2019

Martin NIOMBELA

Léo PIERRE

Table des matières

Introduction	2
Conception générale	2
Diagramme de cas d'utilisation	2
Diagramme de classes	3
Diagramme de séquence	3
Création de frise	3
Création d'événement	4
Chargement d'une frise	4
Conception détaillée	4
La structure centrale : Infofrise	5
Affichage des images et des données détaillées : Diaporama	6
Frise et miniatures des images : PanelFrise	6
Conteneur global et événement : PanelTotal et Controleur	6
Manuel utilisateur	8
Introduction	8
Fenêtre initiale	8
Barre de menus	9
Formulaire de saisie d'événement	10
Frise chronologique	11
Conclusion	11

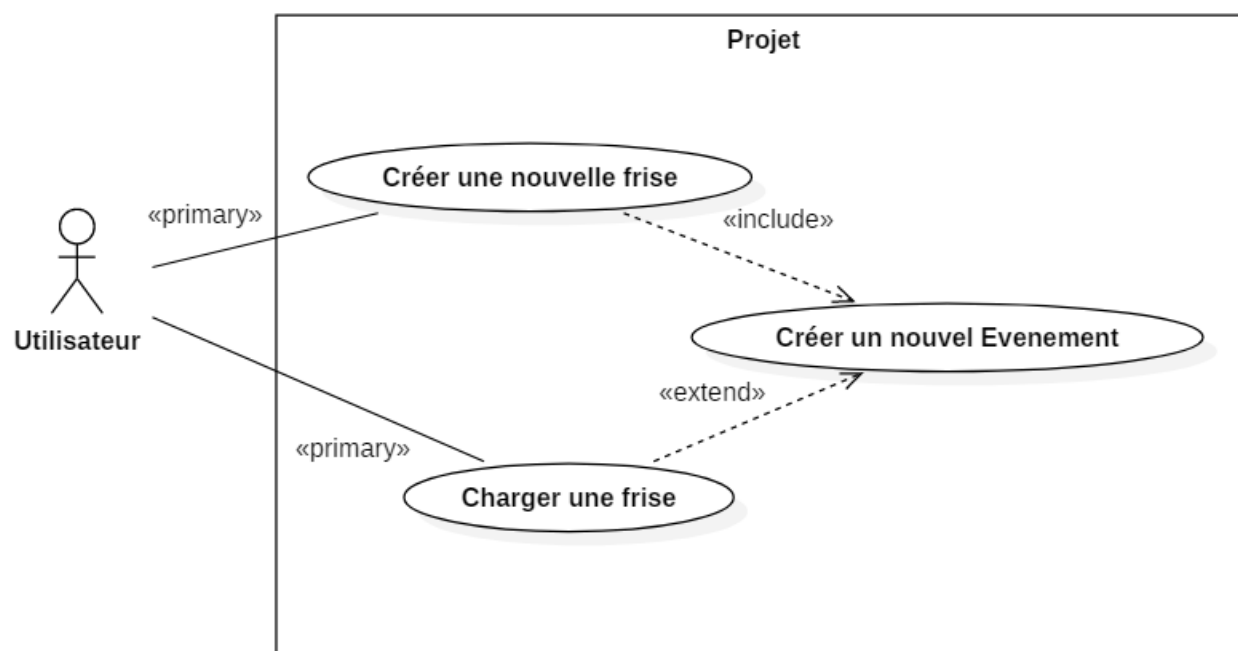
Introduction

L'objectif de ce projet est de réaliser une application permettant à l'utilisateur de créer ou charger une frise. L'application pourra ajouter des événements à la frise et permettra de les visualiser par une fenêtre affichant un Diaporama d'événements et une Frise.

Ce rapport comporte la conception générale, la conception détaillée et le guide d'utilisateur dans l'ordre cité.

Conception générale

Diagramme de cas d'utilisation



Le **diagramme de cas d'utilisation** liste et permet de visualiser plus facilement les différentes situations pouvant se présenter lors de l'utilisation de l'application. Dans notre cas, l'utilisateur peut :

- Créer une nouvelle frise
- Charger une frise existante
- Créer un nouvel Événement

Lorsque l'utilisateur crée une nouvelle frise, il passe obligatoirement par la création d'un nouvel événement. Lorsque l'utilisateur charge une frise, il peut soit y ajouter un événement, soit visualiser la frise telle qu'elle était à sa dernière sauvegarde. Ainsi, l'ajout d'un nouvel événement est obligatoire lors de la création d'une nouvelle frise, mais est optionnel après le chargement d'une frise déjà créée.

Diagramme de classes

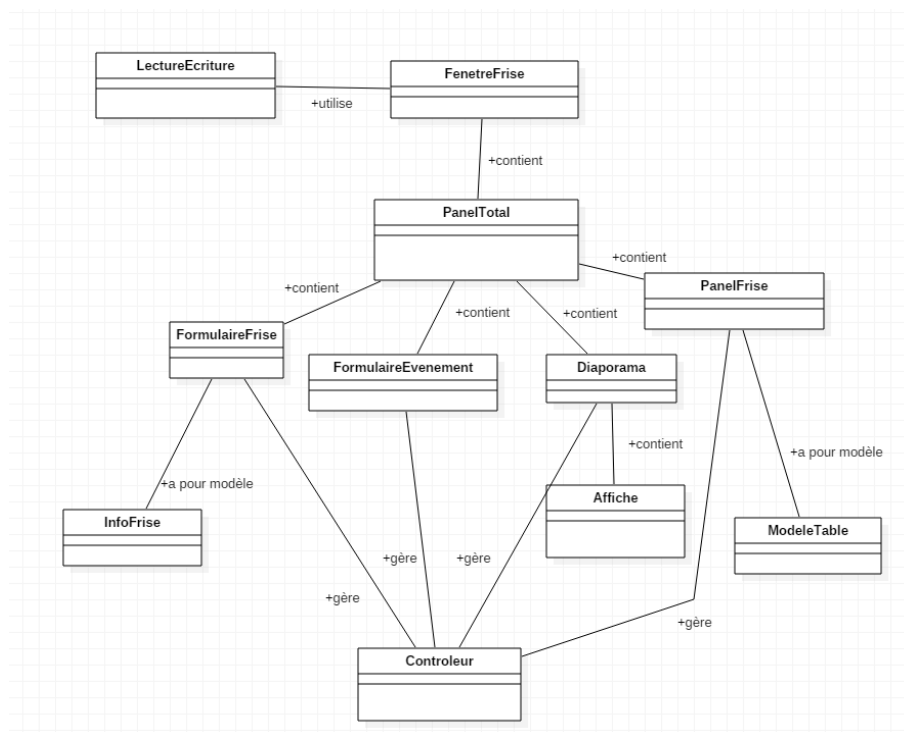
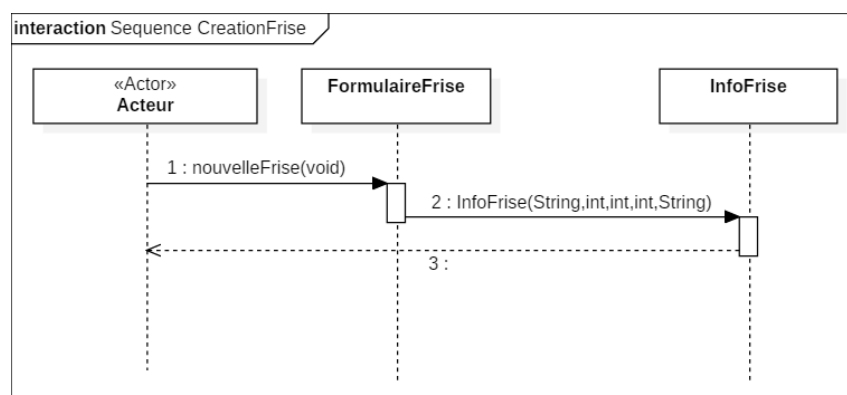
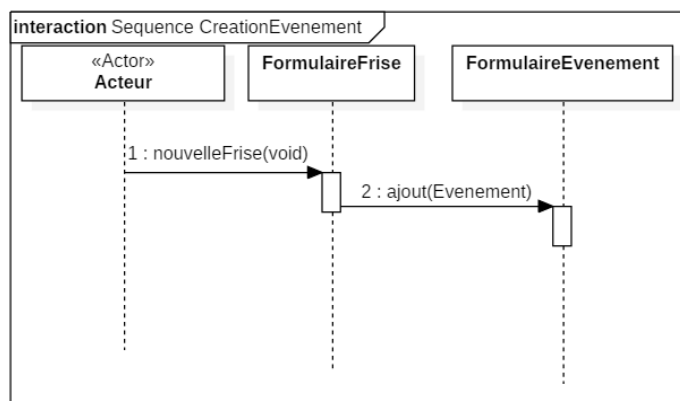


Diagramme de séquence

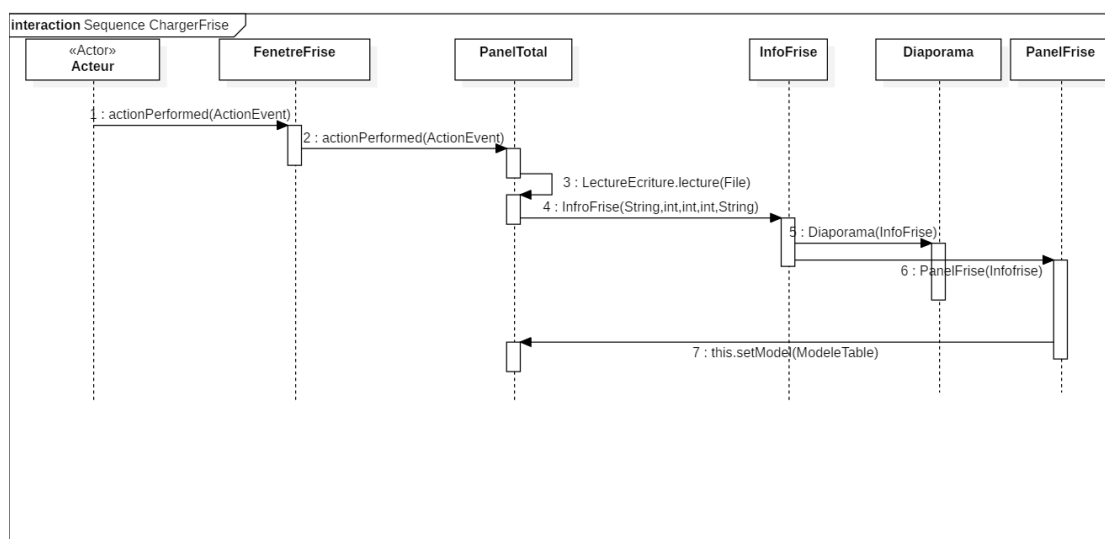
Création de frise



Création d'événement



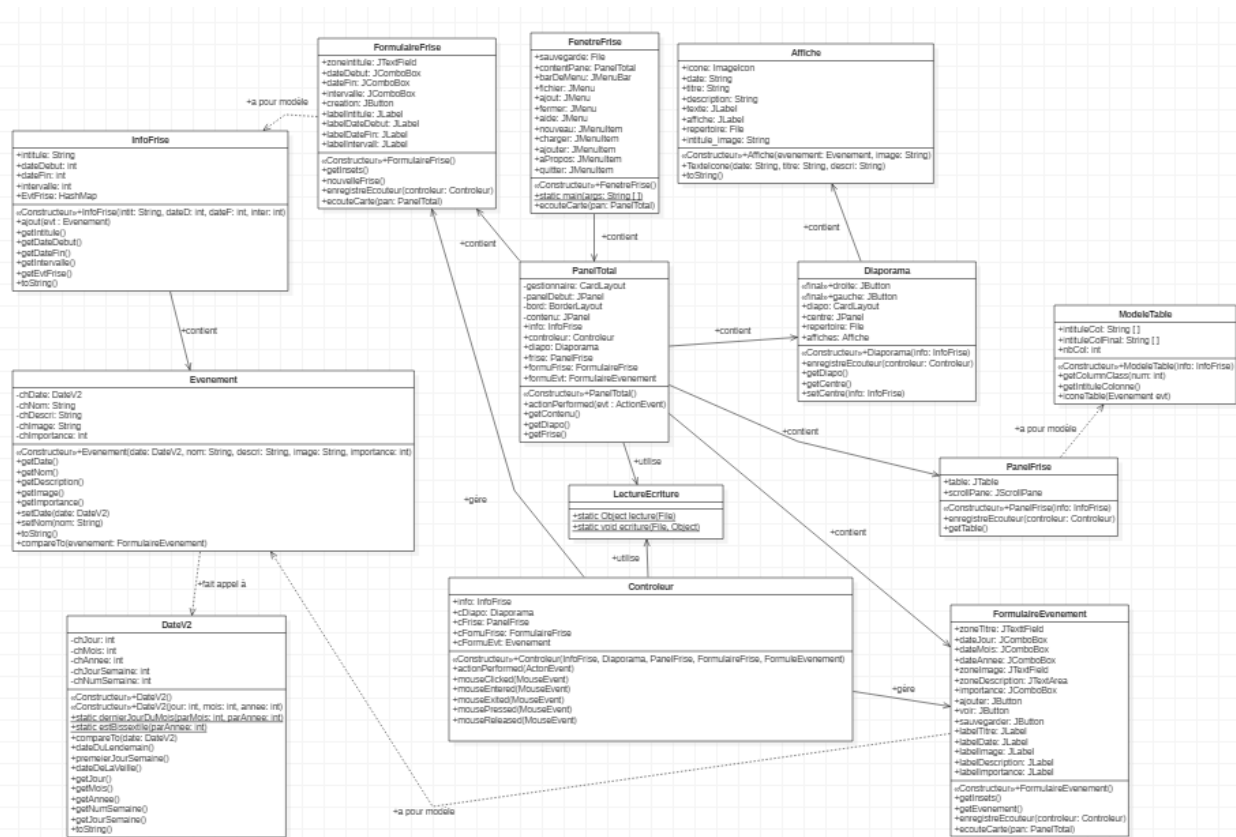
Chargement d'une frise



Conception détaillée

Le projet Frise est composé de différentes classes, qui sont organisées telles que :

Package modele	Package vue	Package controleur
<ul style="list-style-type: none"> • DateV2 • Evenement • InfoFrise • LectureEcriture • ModeleTable 	<ul style="list-style-type: none"> • Affiche • Diaporama • FenetreFrise • FormulaireEvenement • FormulaireFrise • PanelFrise • PanelTotal 	<ul style="list-style-type: none"> • Controleur



La structure centrale : Infofrise

Pour pouvoir afficher correctement la frise, nous avons décidé d'une certaine structure pour notre programme. Ainsi, à la création de la crise, nous avons besoin d'une classe qui contient l'intitulé de la frise, sa date de début et de fin, et l'intervalle d'affichage des années dans la `JTable`. C'est notre classe `InfoFrise`. De plus, cette classe possède une `HashMap` ayant pour clé des dates, et pour valeur des `TreeSet` d'événement pour chaque date.

Pour remplir les champs de la classe `InfoFrise`, nous avons 2 formulaires : `FormulaireFrise` et `FormulaireEvenement`. Le `FormulaireFrise` sert à remplir les 4 premiers champs de la classe `InfoFrise` (intitulé, date de début, date de fin, intervalle), tandis que le `FormulaireEvenement` sert à remplir la `HashMap` de la classe `InfoFrise`.

Quand `InfoFrise` possède des valeurs (donc qu'elle est différente de `null`), 2 classes sont appelées et utilisent les informations que contiennent `InfoFrise`, `Diaporama` et `PanelFrise`. `Diaporama` et `PanelFrise` sont des classes du package `vue`.

Affichage des images et des données détaillées : **Diaporama**

Prenons d'abord **Diaporama**. **Diaporama** est un **BorderLayout** possédant en son centre un panel et sur ses côtés est et ouest des boutons. Le panel au centre de **Diaporama** est un **CardLayout** qui contient des instances de la classe **Affiche**. **Affiche** est une classe du package **vue**. C'est un **JPanel** qui contient l'image, le titre, la date et la description d'un événement qui lui est passé en paramètre. Ce **JPanel** est un **BorderLayout** contenant 2 **JLabel**, un label pour l'image et un autre pour le titre, la date et la description mis sous format html.

Diaporama, qui va parcourir la **HashMap** de la classe **InfoFrise**, va créer pour chaque événement une affiche et stocker cette affiche nouvellement créée dans un tableau d'instance de la classe **Affiche**, un des champs de la classe **Diaporama**.


Frise et miniatures des images : **PanelFrise**

PanelFrise est une classe du package **vue**. C'est la classe qui va afficher la **JTable** qui nous servira de frise chronologique. Cette classe se sert de la classe **ModeleTable** du package **modele** comme modèle. **ModeleTable** reçoit **InfoFrise** en paramètre et se sert de ses dates de début et de fin pour créer sa table et ses intitulés de colonnes. Elle se sert aussi de la classe **InfoFrise** pour mettre les événements à la dans la bonne case de la table. Les classes **Diaporama** et **PanelFrise** ainsi instanciées sont ensuite placées dans le **PanelTotal**.

Conteneur global et événement : **PanelTotal** et **Controleur**

Le **PanelTotal** est une classe du package **vue**. C'est un **CardLayout** qui contient les classes **Diaporama**, **PanelFrise**, **FormulaireEvenement** et **FormulaireFrise**. **Diaporama** et **PanelFrise** sont contenues dans un même **JPanel**, contenu, qui est un **BorderLayout**. Le passage d'une carte à une autre de la classe **PanelTotal** est géré par la **JMenuBar** de la **FenêtreFrise**.

La gestion des événements de la fenêtre se fait dans 2 classes : **Controleur** et **PanelTotal**. Pour des raisons de visibilité, nous avons préféré scinder le contrôle des objets et le contrôle des différents panels dans ces 2 classes. Ainsi, toutes les actions qui



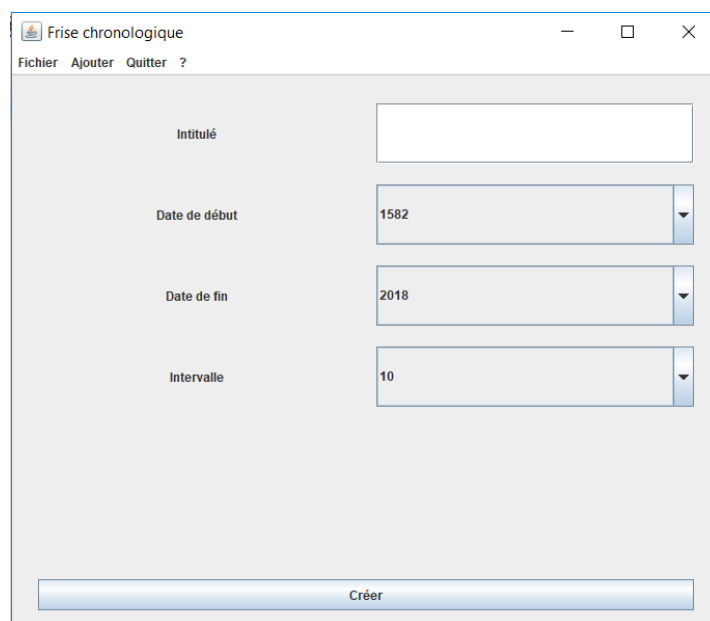
concernent l'affichage d'un panel sont implémentées dans l'actionPerformed du `PanelTotal`. Le reste des actions est géré par le `Contrôleur`. Ce dernier se doit d'avoir accès à la plupart des classes. Il reçoit donc en paramètre l'`InfoFrise`, le `PanelFrise`, le `Diaporama`, et les 2 formulaires.

Manuel utilisateur

Introduction

L'application « **Frise Chronologique** » est un outil permettant de créer et de visualiser une frise chronologique, tout en y ajoutant des événements en direct. Il est également possible pour l'utilisateur de charger une frise créée préalablement, pour la visualiser et éventuellement la modifier.

Fenêtre initiale



Lorsque l'utilisateur ouvre l'application, il arrive sur la fenêtre ci-dessus. Cette fenêtre est le formulaire de saisie d'une frise. Ainsi, par défaut, l'utilisateur a la possibilité de créer une nouvelle frise avec une date de début, une date de fin et un intervalle. Pour créer une nouvelle frise, l'utilisateur doit remplir les 4 champs suivants :

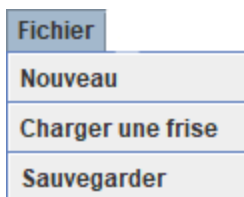
- **Intitulé** (Nom de la frise)
- **Date de début** (Début de la frise)
- **Date de fin** (Fin de la frise)
- **Intervalle** (Intervalle entre chaque numéro d'année affiché)

La fenêtre comporte également le bouton **Créer** pour passer à l'étape suivante ([Formulaire de saisie d'événement](#)).

Barre de menus

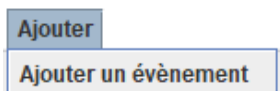
Il est également possible d'utiliser la barre de menus située en haut à gauche de la fenêtre et qui comporte les menus **Fichier**, **Ajouter**, **Quitter** et **?**.

Le menu **Fichier** contient :



- **Nouveau** : Créer une nouvelle frise
- **Charger une frise** : Charger une frise déjà créée
- **Sauvegarder** : Redirige vers le [FormulaireEvenement](#)

Le menu **Ajouter** contient :



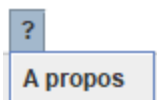
- **Ajouter un évènement** : Ouvre le formulaire de saisie d'évènement

Le menu **Quitter** contient :

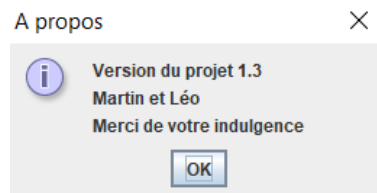


- **Quitter la fenêtre** : Ferme la fenêtre et l'application

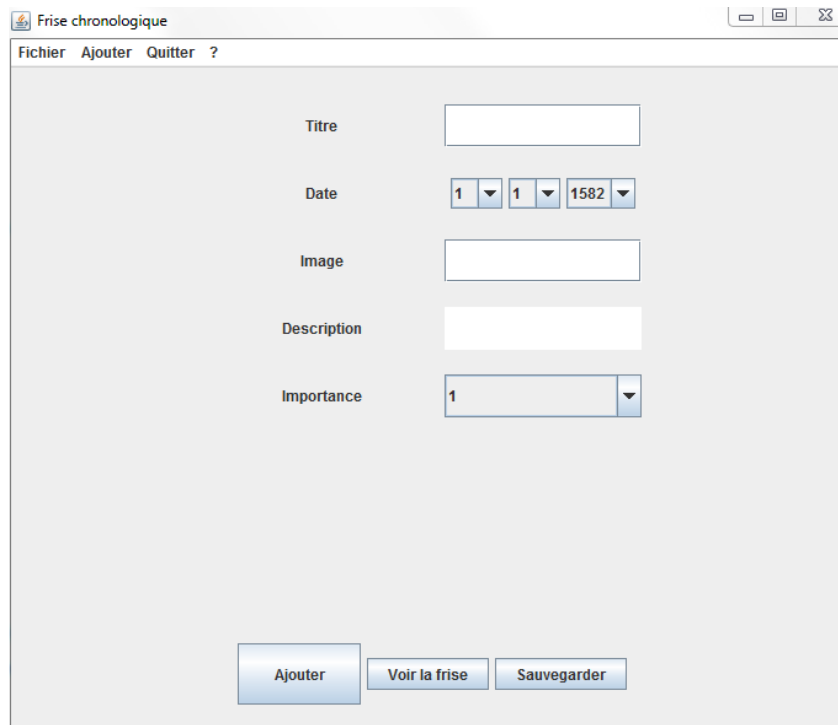
Le menu **?** contient :



- **À propos** : Ouvre la fenêtre suivante



Formulaire de saisie d'événement



The screenshot shows a window titled 'Frise chronologique' with a menu bar containing 'Fichier', 'Ajouter', 'Quitter', and '?'. The main area contains five input fields: 'Titre' (text), 'Date' (three dropdowns for day, month, and year, with values 1, 1, and 1582 respectively), 'Image' (text), 'Description' (text), and 'Importance' (dropdown with value 1). At the bottom are three buttons: 'Ajouter', 'Voir la frise', and 'Sauvegarder'.

Lorsque l'utilisateur appuie sur le bouton **Créer** dans le formulaire de frise ou s'il clique sur **Ajouter**, il arrive sur la fenêtre ci-dessus. Cette fenêtre est le formulaire de saisie d'un événement, dans lequel l'utilisateur doit remplir les 5 champs suivants :

- **Titre** (Titre de l'événement)
- **Date** (Date de l'événement, Jour, Mois, Année)
- **Image** (Image représentant l'événement)
- **Description** (Description de l'événement)
- **Importance** (Niveau d'affichage, plus ou moins proche, du Diaporama)

La fenêtre comporte également les boutons suivants :

- **Ajouter** (Ajout de l'événement à la frise)
- **Voir la frise** (Affichage de la frise)
- **Sauvegarder** (Sauvegarde des événements ajoutés)

Frise chronologique

En cliquant sur *"Voir la frise"* ou en chargeant une frise existante, la fenêtre **Frise Chronologique** s'ouvre. Elle est composée du **Diaporama** et de la **Frise**.

Pour parcourir le diaporama, l'utilisateur doit utiliser les deux boutons situés à gauche et à droite du **Diaporama**. Le **Diaporama** est, lui, composé de l'image de l'événement, sa date précise, son titre et sa description.

Pour parcourir la frise, il faut utiliser la barre horizontale située en dessous de cette dernière.

Conclusion

Le projet est presque totalement fonctionnel.

Nous pensons que nous aurions pu finir le projet avec un peu plus de temps. En effet, si la sauvegarde fonctionne, nous n'avons pas réussi à faire en sorte que le chargement d'une frise se fasse correctement. Pour gagner du temps, nous avons aussi dû renoncer à la robustesse de notre programme et prendre quelques raccourcis au niveau du fonctionnement de notre interface.

La quantité de travail était non négligeable et nous avons une faible connaissance de l'outil Git. Il nous manque le dossier de test et la Javadoc, dont nous avons été dans l'impossibilité de générer.

La conception reste quand même l'étape qui nous a pris le plus de temps, car nous ne savions pas vraiment par où commencer. De plus, même si cela est de moindre importance, l'esthétique de notre programme est peu, voire pas du tout travaillée.

Cependant, nous pensons que notre programme répond quand même à pas mal de critères, malgré le laps de temps qui nous a été donné. Nous nous sommes investis dans notre projet, comme peuvent en témoigner les nombreux commits de notre dépôt bitbucket.