BROSSARD Florian / MILLOTTE Fanny

Projet tuteuré de 3ème année en licence d'informatique

Logiciel d'aide à la réalisation d'algorithmes et génération de code LaTeX

Tuteur de projet : M. BOUQUET Fabrice

UFR ST

16 route de Gray
25000 Besançon

Remerciements

Nous remercions Monsieur BOUQUET pour ses précisions sur le sujet, ses conseils, ses idées d'améliorations et pour sa disponibilité depuis le début du projet. Nous le remercions également pour la rapidité et la clarté avec lesquelles il à su répondre aux questions que nous nous sommes posés tout au long de ce projet.

Table des matières

1 - Présentation

1.1 - Contexte

Ce projet a été réalisé durant notre troisième année en licence d'informatique à l'UFR ST de Besançon. Il fait partie des épreuves comptabilisées pour le semestre six de notre cursus et est à réalisé en binôme.

1.2 - Présentation du sujet

Le but de ce projet est de réaliser un logiciel aidant à la réalisation d'un diaporama représentant un algorithme pour cela le logiciel doit générer du code LaTeX. Le choix du langage de programmation étant libre.

Nous avions une représentation du rendu visuel d'une diapositive (voir ci-dessous). Nous pouvons ainsi observer le pseudo code en haut à gauche, le tableau des variables en haut à droite, de même que le graphe représentant l'avancement de l'algorithme en bas.

Le code LaTeX généré par le logiciel doit être basé sur beamer pour la gestion des animations ainsi que Tikz pour la gestion du graphe.

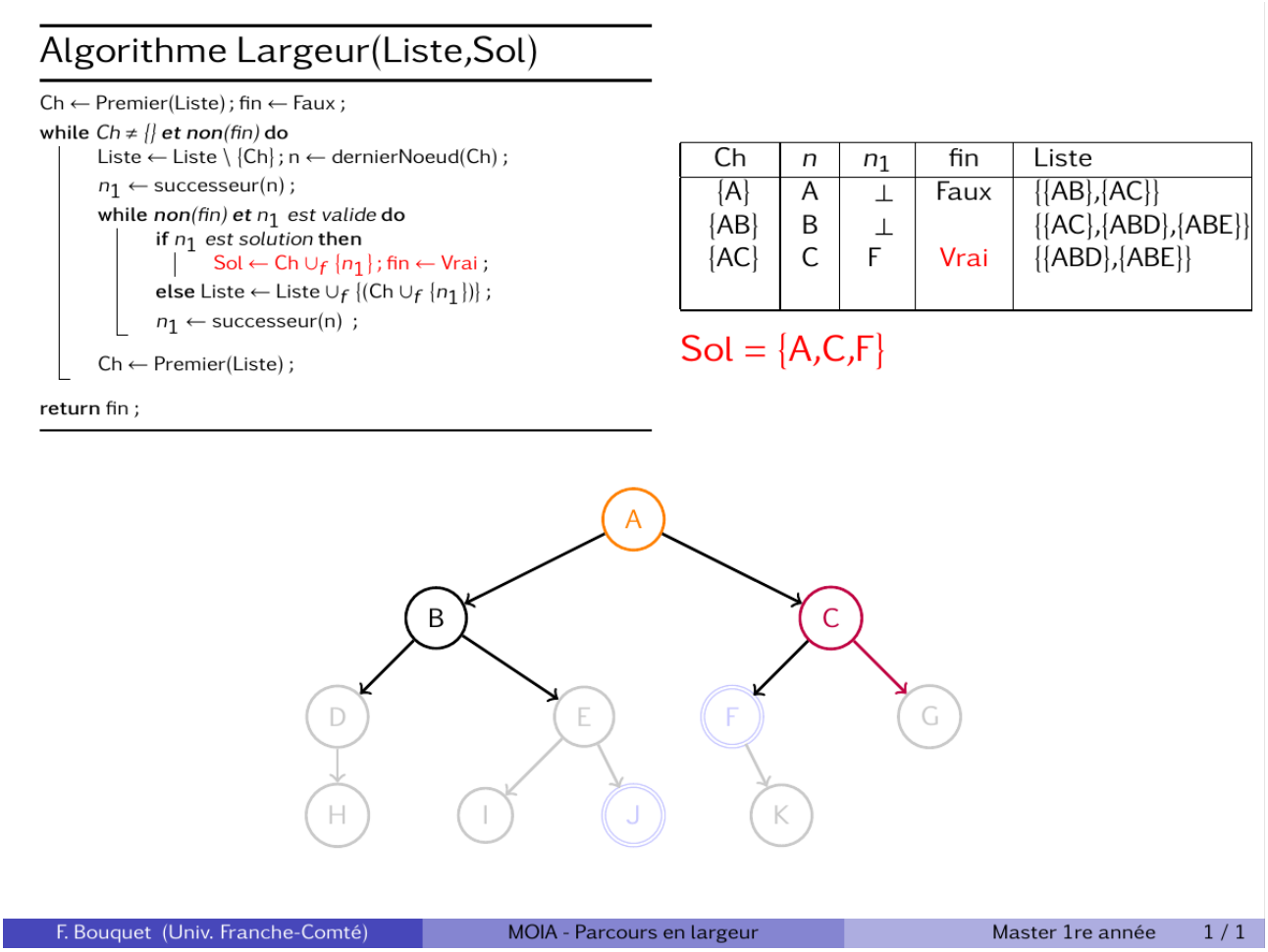


Illustration 1 - Exemple d'une diapositive

2 - Début du projet

Nous avions deux choix pour effectuer ce projet, dans un premier temps reprendre le travail réalisé durant l'année 2015 – 2016 par un autre binôme ou recommencer le projet du début.

2.1 - Première solution

Nous nous sommes d'abord tourné vers le premier choix, reprendre le travail laisser par un autre binôme. Le binôme précédent a choisi de réaliser le projet en C++ et Qt. Nous avons rencontrés de nombreuses difficultés tout d'abord avec la compréhension du code, nous n'avions fait que très peu de C++ et nous n'avions jamais utilisé Qt. L'absence de nombreux commentaires, de documentation et la non-organisation des fichiers a entravée

grandement notre travail de compréhension du code.

Le projet de ce binôme était presque entièrement terminé il ne manquais que les tests unitaires.

Le faite d'avoir essayé de reprendre ce projet nous à appris beaucoup de chose sur le code à laisser à d'éventuels personnes. Tout d'abord l'importance d'avoir un code clair et de ne pas oublier de commenter ce code tout au long du développement ainsi que l'importance de la documentation à apporter à ce code.

N'étant pas suffisamment à l'aise avec le langage C++ et n'ayant pas beaucoup avancé dans la compréhension du code pendant une période d'un mois nous nous sommes tourné vers la deuxième solution.

2.2 - Seconde solution

Nous avons finalement opté pour la seconde solution recommencer le projet du début tout en intégrant ce que nous avons appris en essayant la première solution. Nous souhaitons éviter que notre programme soit trop difficile à reprendre si un éventuel binôme le fait.

Nous voulons choisir un langage que nous maîtrisons mieux, le C++ étant un langage que nous avons peu expérimenté.

3 - Analyse des besoins

Une analyse des besoins est nécessaire à la compréhension du projet. Elle permet de définir ce qui est essentiel au projet et ce qui est optionnel. Ceci nous a aidé à définir dans quel ordre réaliser les différentes fonctionnalités de notre logiciel.

3.1 - Besoins de l'utilisateur

L'utilisateur a besoin de définir le déroulement de l'algorithme pas à pas à chaque diapositive. Donc le logiciel doit disposer d'un éditeur pour le pseudo code ainsi qu'une mise en évidence sur ce dernier.

L'utilisateur a besoin de montrer clairement les valeurs contenues dans les variables définies dans le pseudo code. Il doit également pouvoir montrer à quel moment une variable

change de valeur. Donc le logiciel doit contenir un tableau représentant les variables du pseudo code ainsi qu'une mise en évidence des variables au moment d'un changement de valeur.

L'utilisateur a besoin d'éditer un graphe représentant le déroulement de l'algorithme pas à pas ainsi qu'une mise en évidence du chemin parcouru à un moment précis. Le logiciel doit donc comporter une partie permettant l'édition d'un graphe ainsi qu'une mise en évidence d'une partie précise du graphe représentant le chemin qui est parcouru.

L'utilisateur à également besoin de visualiser le rendu diapositive par diapositive. Il est donc impératif de pouvoir effectuer un changement de diapositive afin de visualiser l'avancement ou le rendu final d'un diaporama.

Pour finir le logiciel doit retranscrire les informations saisies par l'utilisateur sous la forme d'un code LaTeX.

3.2 - Aspect visuel

Pour une meilleur vision du rendu final d'un diaporama nous souhaitons que l'interface de notre logiciel soit le plus proche possible de la représentation d'une diapositive. Nous souhaitons également rendre l'aspect visuel du logiciel proche des logiciels d'édition couramment utilisés tel que les éditeurs de textes, de tableaux, etc.

Ainsi cinq parties seront présentes lors de l'édition d'un diaporama.

Dans un premier temps, un menu permettant l'ajout d'un nouveau diaporama, sa sauvegarde ainsi que la fermeture du logiciel. De même que L'ajout et la suppression des colonnes et lignes du tableau correspondant aux variables. Ainsi qu'une rubrique d'aide à la manipulation du logiciel.

La deuxième partie est constitué d'une barre d'outils repositionnable par l'utilisateur. Cette barre d'outils comprend les boutons contrôlant le changement de diapositive ainsi que l'édition du graphe représentant l'évolution de l'algorithme.

Nous en venons à la partie permettant l'édition du pseudo code. Nous souhaitons la faire ressembler à un éditeur de texte standard comprenant les numéros des lignes puis la mise en évidence d'une ligne par une puce correspondant à côté du numéro de la ligne ainsi que le pseudo code.

Ensuite, dans la quatrième partie, nous retrouvons le tableau permettant la représentation des valeurs des différentes variables du pseudo code. Le changement de la valeur d'une variable d'une diapositive à une autre entraînant le mise en évidence de cette

valeur.

Pour finir, la cinquième partie correspond au graphe représentant le déroulement de l'algorithme. Le chemin suivit durant le déroulement de l'algorithme est mis en évidence par le changement de couleur de la partie correspondante du graphe. Le parcours du graphe est visible par l'apparition des zones parcourus précédemment par l'algorithme. Les états finaux sont représentés par un double cercle et les états non accessible par un cercle barré.

3.3 - Aspect technologique

Pour la réalisation de ce projet nous avons choisi d'utiliser JAVA. C'est le langage objet que nous avons le plus utilisé et nous avions également une matière lors de notre 5ème semestre qui nous apprend comment utiliser la bibliothèque graphique swing de java. Cette matière à pu nous aider pour la réalisation de ce projet de même que le projet nous a aider pour l'élaboration du projet de cette matière.

L'implémentation de tests unitaire nous permettra de déceler d'éventuels erreur durant le développement de notre logiciel et nous aidera à améliorer la structure de ce dernier.

Afin garder une trace des précédentes versions de notre logiciel et de faciliter le travail en binôme, nous avons choisi d'utiliser le logiciel de gestion de versions Git que nous n'avions jamais utilisé auparavant.

4 - Conception

4.1 - Structure du logiciel

Pour ce projet nous avons choisi d'utiliser l'architecture Modèle-Vue-Contrôleur pour l'implémentation des classes. Les diagrammes de classe correspondant aux différentes parties du projet sont présent en annexe.

4.2 - Conception des différents éléments

4.2.1 - Changement de diapositive

Le changement de diapositive s'effectue à l'aide de boutons présent dans la barre d'outils, on peu également passer à une diapositive précise en indiquant son numéro via un champ de texte présent dans la barre d'outils.

Il est également possible de ne changer de diapositive uniquement pour les parties souhaitées, l'éditeur du pseudo code, le tableau des variables ou le graphe, en sélectionnant cette ou ces parties à l'aide de checkboxs.

4.2.2 - Editeur du pseudo code

L'éditeur de pseudo code est composé de trois éléments. D'abord une zone de saisie de texte où pourra être entré le pseudo code. Puis une colonne de labels comportant le numéro des lignes du pseudo code. La troisième partie est une colonne correspondant au ligne où il est possible d'ajouter des puces pour indiquer quelle partie du texte sera mise en évidence sur la diapositive courante. Le pseudo code est commun entre les différentes diapositives. Il n'est pas possible de mettre en évidence une ligne vide, c'est-à-dire une ligne ne contenant aucun caractère (caractère non-imprimable comme le saut de ligne inclus).

4.2.3 - Tableau des variables

Il est possible, sur le tableau des variables, de changer le nombre de ligne et de colonne ainsi que d'éditer le contenu des cases du tableau. Le contenu de la première ligne du tableau ainsi que les dimensions de ce dernier sont communs entre les diapositives, la première ligne étant le noms des colonnes. Le contenu ajouté ou modifié dans une diapositive est mis en évidence et repris dans les diapositives suivantes.

4.2.4 - Graphe

L'ajout d'un nœud du graphe, d'un lien entre deux nœuds et la suppression de ces deux éléments se fait depuis la barre de menu ou depuis un menu contextuel directement sur les éléments. Il est possible de changer le nom d'un nœud ainsi que celui d'un lien via le

champ de texte qui lui est attribué. Chaque nœud peux bouger indépendamment des autres nœuds cela permet à l'utilisateur de pouvoir ordonner son graphique comme il l'entend. Il est également possible de changer la couleur des nœuds et des liens, permettant de faire une mise en évidence du chemin parcouru par l'algorithme.

5 - Réalisation

5.1 - Etapes du développement

Nous avons débuté notre projet en structurant notre fenêtre pour quelle ai un aspect le plus proche du rendu final attendu. Pour cela nous avons placé en haut à gauche un panel correspondant à l'éditeur de pseudo code, en haut à droite un panel correspondant au tableau des variables et en bas un panel qui contiendra la partie du graphe.

Par la suite nous avons implémenté notre éditeur de pseudo code. Dans un premier temps nous avons ajouter une simple zone de texte et nous nous sommes penché sur la problématique suivante : Comment visualiser la mise en évidence d'une ou de plusieurs lignes du texte ? Nous souhaitions nous rapprocher le plus des outils informatique utilisé dans la vie courante. Nous avons donc ajouter une colonne à la gauche du texte où est visible le numéro de la ligne et où par un simple clic sur ce numéro, une puce indiquera que la ligne sera mise en évidence pour la diapositive courante.

Puis nous sommes passés à la réalisation de la partie comprenant le tableau des variables. Nous avons tout d'abord créés un tableau où il est possible de supprimer ou d'ajouter des lignes et des colonnes. Puis nous avons ajouter une fonction permettant de mettre en évidence une case modifier durant la diapositive courante par rapport à la précédente.

Nous avons par la suite choisi de tester nos deux premières parties en réalisant les fonctions traitant le changement de diapositives. Pour cela nous avons lié le texte du pseudo code à toutes les diapositives, une modification apporté sur une diapositive est donc répercuté sur toutes les autres, de même pour les noms des colonnes du tableau. Nous considérons que chaque diapositive représente un pas dans l'algorithme, pour cela nous avons donc choisi de faire passé à la ligne suivante la puce du pseudo code lors de la création d'une nouvelle diapositive.

Sauvegarde des données

Partie graphique

5.2 - Problème rencontrés

5.3 - Améliorations effectués

6 - Bilan

6.1 - Bilan pédagogique

Ce projet nous à permis d'utiliser les compétences que nous avons acquis durant notre cursus universitaire et de parfaire ces connaissances. Cela nous à notamment aider lors des projets et examens des différentes matières liés à java swing pour l'interface graphique, à XML pour la sauvegarde et aux arbres de syntaxe abstraite pour l'édition du pseudo code.

Nous avons également pu expérimenté l'aspect programmeur / client. Comment répondre aux besoins d'un client, déterminer ce qui est essentiel à ce client pour la création du projet. Cela nous sera très utile pour notre futur stage en entreprise.

6.2 - Bilan technique

Les fonctionnalités que nous avons déterminé comme essentiel ont été implémentés la partie gérant le pseudo code permet d'indiquer sur quelles lignes mettre la mise en évidence, par une puce lier à la ligne, en fonction de la diapositive courante, l'ajout d'une diapositive fait passer la puce à la ligne suivante. Le changement d'une partie du pseudo code est répercuté sur les autres diapositive.

La partie tableau permet l'ajout ou la suppression de lignes et de colonnes les titres sont communs à toutes les diapositives et la mise en évidence des données ajoutées ou modifiées est présente.

La partie graphique

Le changement de diapositives

La sauvegarde

la génération du code LaTeX

L’arbre de syntaxe abstrait

6.3 - Bilan général

Cahier des charges rempli, application fonctionnelle, amélioration effectuer en plus, problème rencontré résolu, …

6.4 - Améliorations possibles

Comme le nom l’indique

7 - Annexes

A - Diagrammes UML

Diagramme de la partie vue

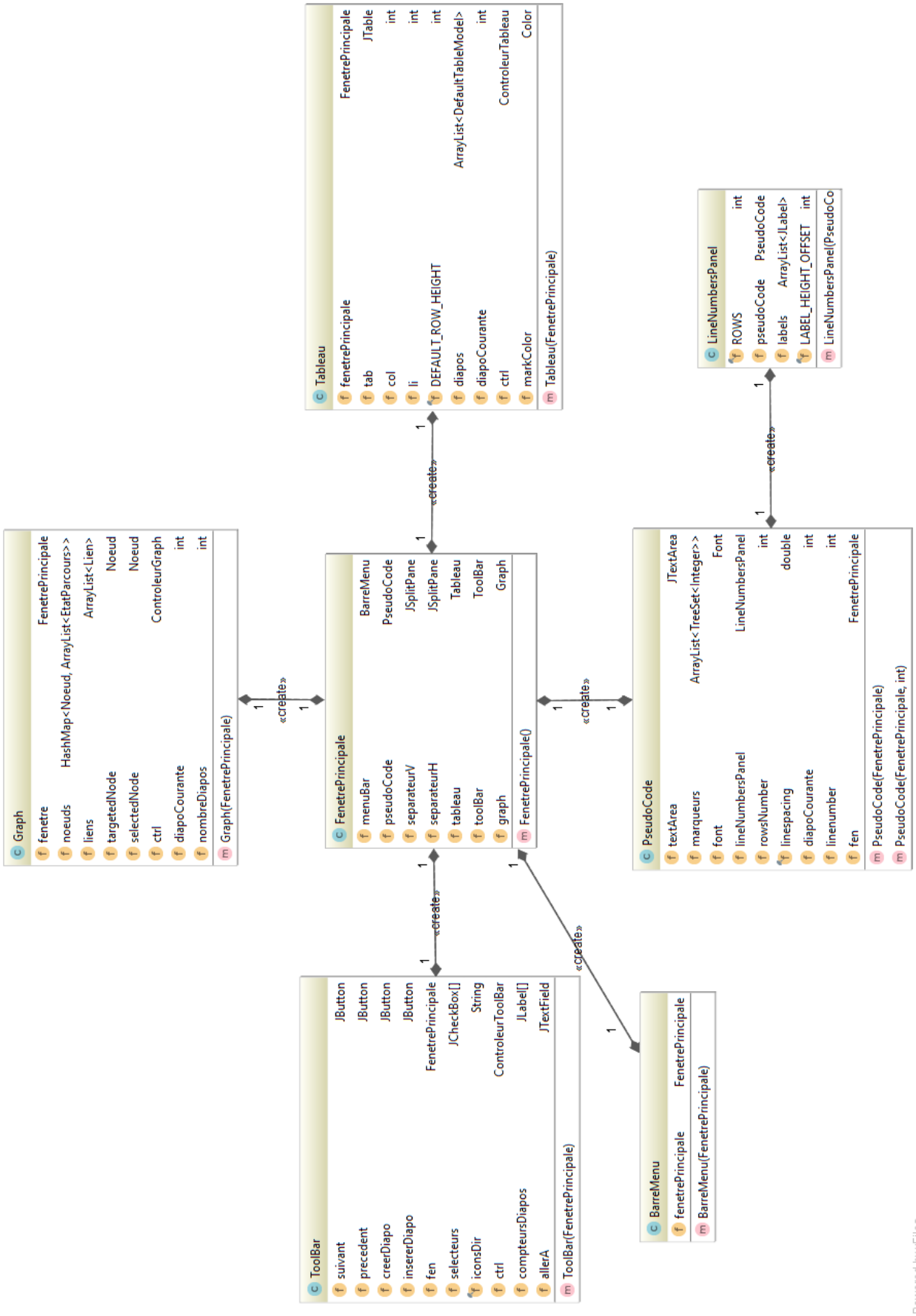


Diagramme de la partie modèle

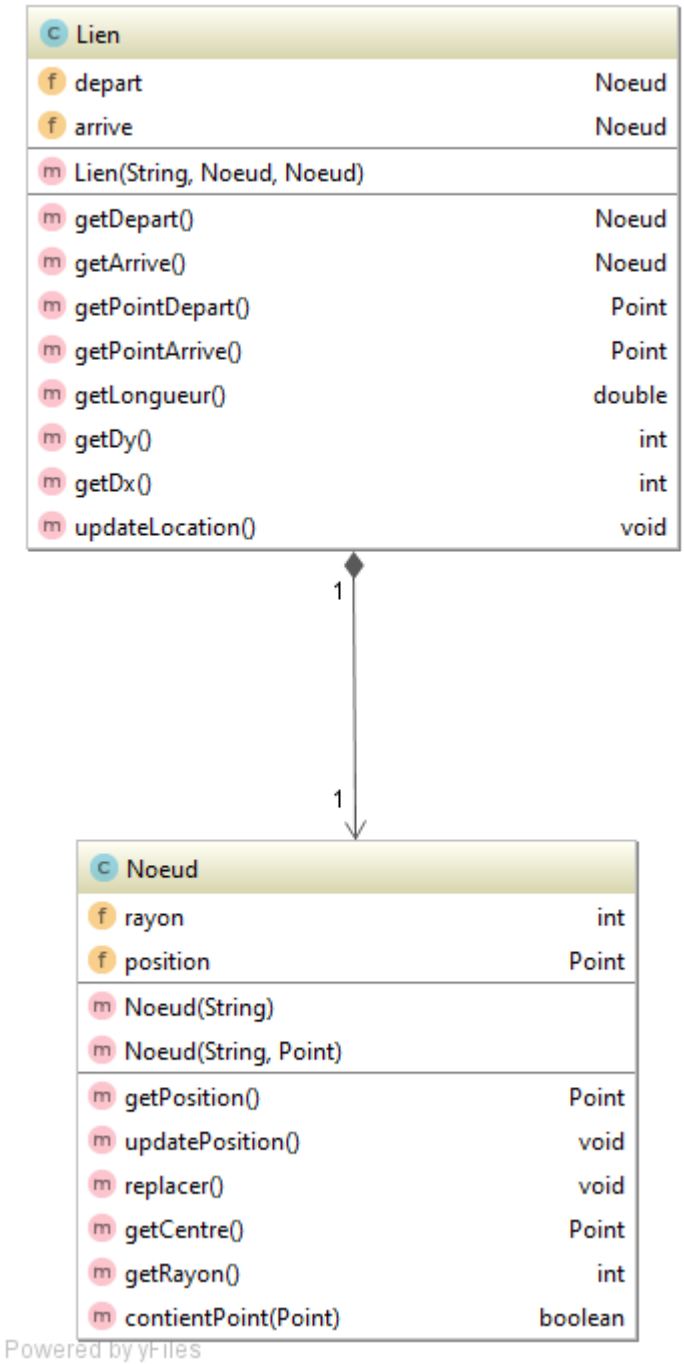


Diagramme de la partie contrôleurs

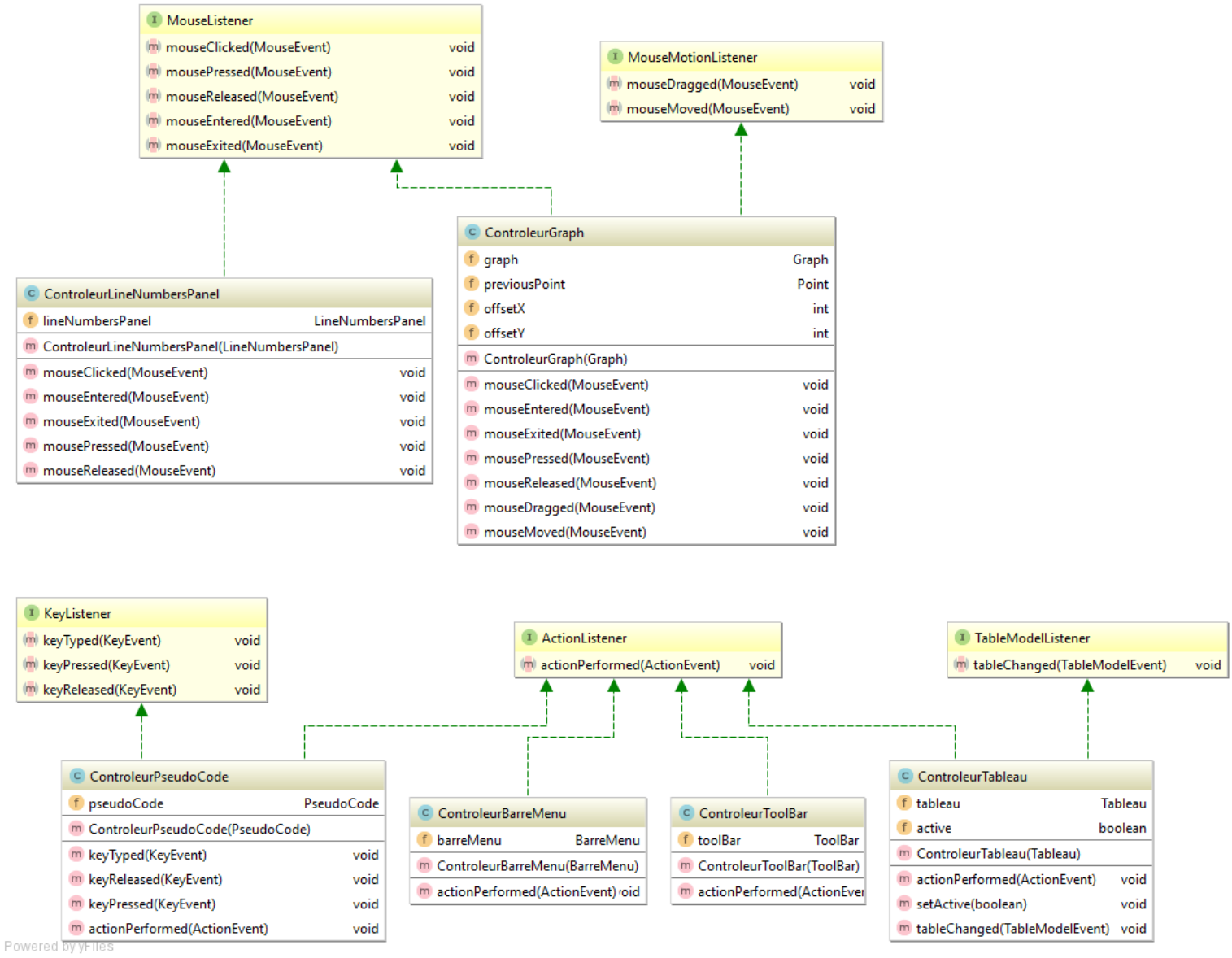
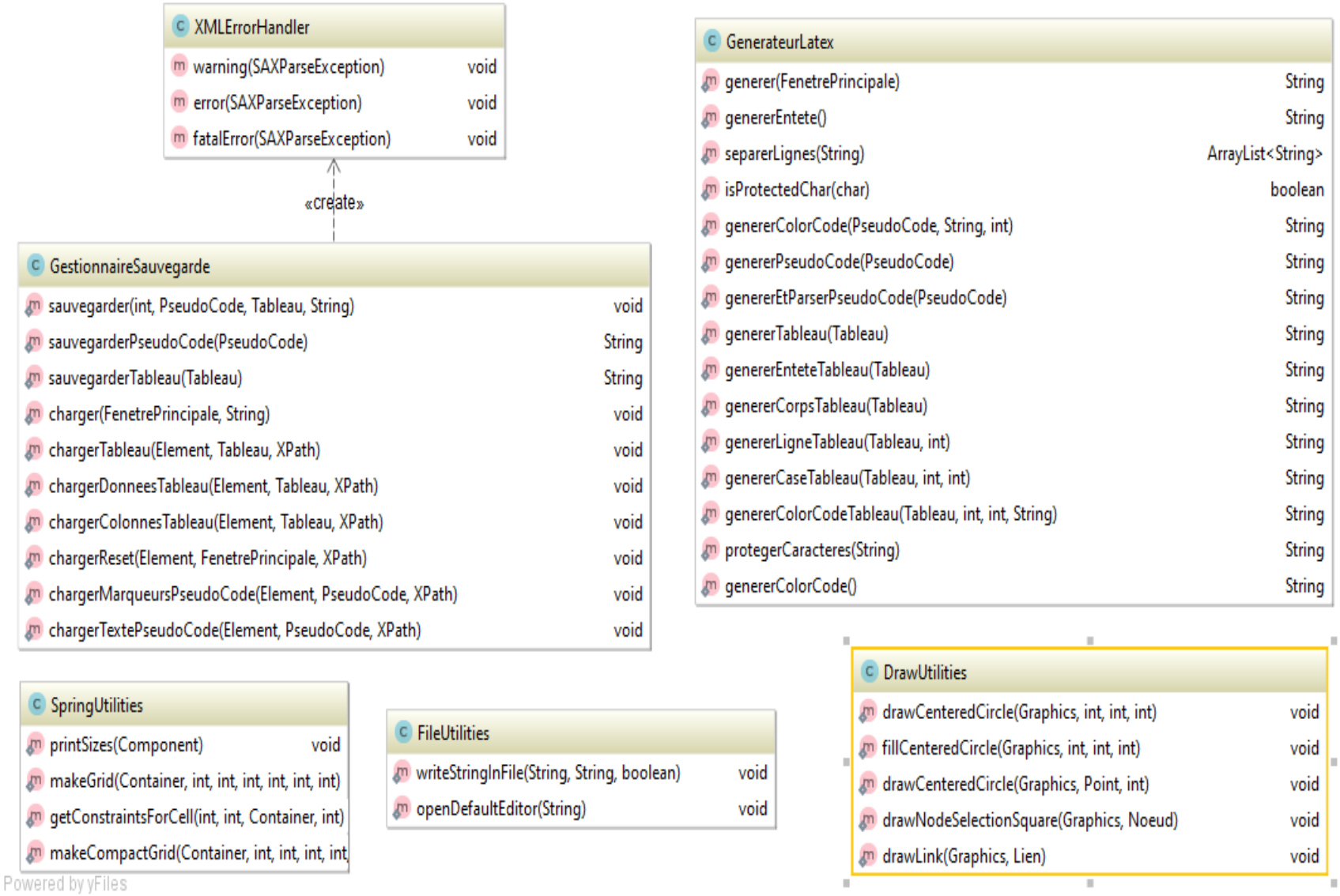


Diagramme des utilitaires


XMLErrorHandler
warning(SAXParseException) void
error(SAXParseException) void
fatalError(SAXParseException) void
«create»
GestionnaireSauvegarde
sauvegarder(int, PseudoCode, Tableau, String) void
sauvegarderPseudoCode(PseudoCode) String
sauvegarderTableau(Tableau) String
charger(FenetrePrincipale, String) void
chargerTableau(Element, Tableau, XPath) void
chargerDonneesTableau(Element, Tableau, XPath) void
chargerColonnesTableau(Element, Tableau, XPath) void
chargerReset(Element, FenetrePrincipale, XPath) void
chargerMarqueursPseudoCode(Element, PseudoCode, XPath) void
chargerTextePseudoCode(Element, PseudoCode, XPath) void
GenerateurLatex
generer(FenetrePrincipale) String
genererEntete() String
separerLignes(String) ArrayList<String>
isProtectedChar(char) boolean
genererColorCode(PseudoCode, String, int) String
genererPseudoCode(PseudoCode) String
genererEtParserPseudoCode(PseudoCode) String
genererTableau(Tableau) String
genererEnteteTableau(Tableau) String
genererCorpsTableau(Tableau) String
genererLigneTableau(Tableau, int) String
genererCaseTableau(Tableau, int, int) String
genererColorCodeTableau(Tableau, int, int, String) String
protegerCaracteres(String) String
genererColorCode() String
SpringUtilities
printSizes(Component) void
makeGrid(Container, int, int, int, int, int, int)
getConstraintsForCell(int, int, Container, int)
makeCompactGrid(Container, int, int, int, int,
FileUtilities
writeStringInFile(String, String, boolean) void
openDefaultEditor(String) void
DrawUtilities
drawCenteredCircle(Graphics, int, int, int) void
fillCenteredCircle(Graphics, int, int, int) void
drawCenteredCircle(Graphics, Point, int) void
drawNodeSelectionSquare(Graphics, Noeud) void
drawLink(Graphics, Lien) void
Powered by yFiles