



## Sujet 6 : Chiffage partiel de fichiers

- Nicolas Devenet ([nicolas@devenet.info](mailto:nicolas@devenet.info))
- Valérieane Jean ([jean.valeriane@gmail.com](mailto:jean.valeriane@gmail.com))

## INTRODUCTION

Le sujet demandait la réalisation d'une application graphique, permettant de chiffrer partiellement un document texte.

L'application a été réalisée avec NetBeans en utilisant les composants graphiques Swing.

## UTILISATION

L'utilisateur a le choix entre trois possibilités lors de l'ouverture de l'application.

1. L'onglet « **Crypt a file** » permet de crypter un fichier, en précisant les lignes à chiffrer, le mot de passe à utiliser, et le dossier de destination.  
Deux fichiers sont générés. Le premier contient tout le texte qui ne doit pas être crypté, et un marqueur<sup>1</sup> est ajouté à la fin ; le second contient le texte crypté.
2. L'onglet « **Decrypt a file** » permet de décrypter les fichiers, en indiquant le mot de passe utilisé. Il suffit de sélectionner le premier fichier généré, l'application se chargera de trouver le second fichier crypté.  
Le fichier généré recompose le fichier initial, en y insérant le texte décrypté.
3. L'onglet « **Display a file** » permet à un utilisateur ne connaissant pas le mot de passe d'afficher tout de même le texte non crypté.  
Par défaut, l'extension de fichier autorisée est celle du fichier généré par l'application, mais on peut forcer la lecture de n'importe quel fichier.

## CODE SOURCE

Le code source du projet est disponible sur GitHub, à l'adresse suivante :

<https://github.com/nicolabricot/TP-Crypto-2013>

- Le dossier `dist/` contient la JavaDoc, générée automatiquement depuis NetBeans, ainsi qu'un JAR (et les librairies nécessaires) de l'application.  
Il suffit de lancer `Projet-Crypto.jar` pour lancer l'application.
- Le dossier `doc/` contient la documentation, dont ce rapport. Le sujet, et les sources utilisées sont aussi disponibles.

---

<sup>1</sup> Ce marqueur est un entier correspondant à la ligne pour « séparer » les parties regroupées.

- Le dossier **lib/** contient les librairies utilisées.
- Le dossier **src/** contient les sources commentées.
- Le dossier **tests/** contient des documents textes utilisés pour tester l'application.

## RÉPARTITION DU TRAVAIL

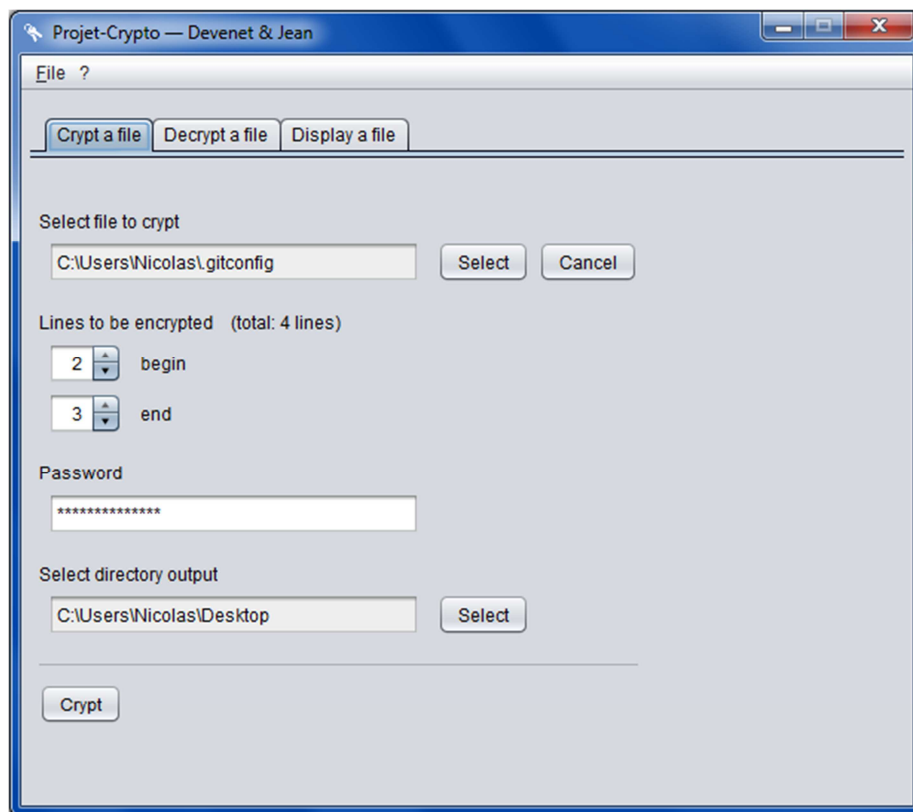
Nous avons réfléchi ensemble à la structure de l'application, et aux solutions à mettre en place pour répondre au sujet demandé.

Nous avons aussi utilisé la classe utilitaire **FileCrypter.java** écrite par Patrick Guichet permettant de crypter et décrypter un fichier, que nous avons modifiée pour nos besoins.

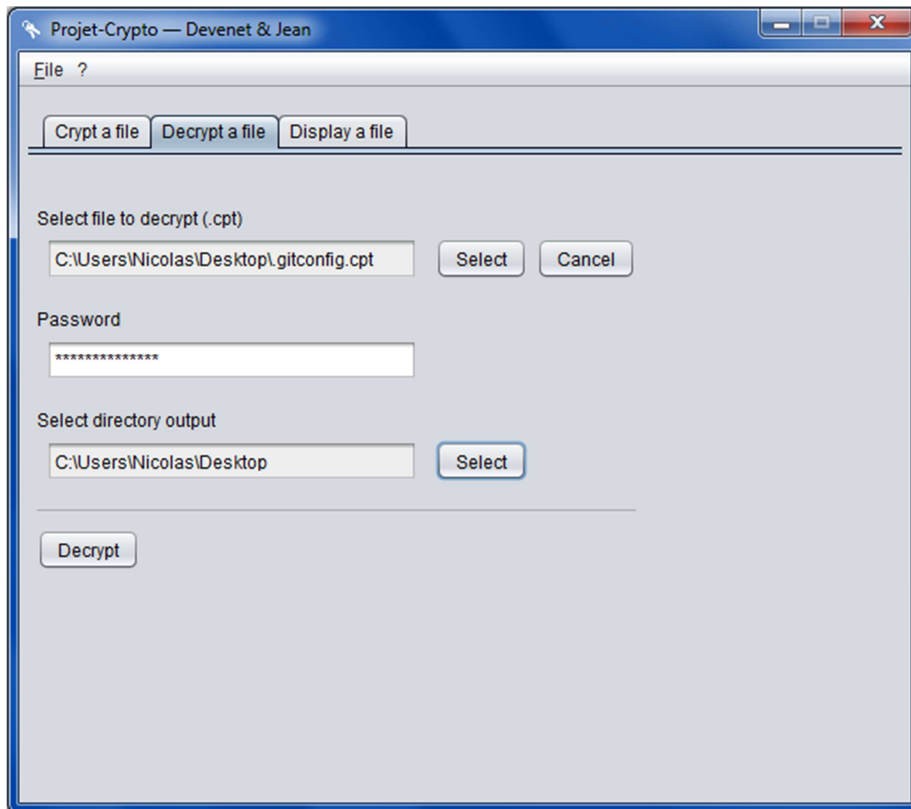
Valériane s'est occupée des bases de l'interface graphique, que Nicolas a ensuite reprises et modifiées pour continuer et finir l'interface. Elle a écrit les fondations des classes pour le cryptage et décryptage.

Nicolas s'est chargé d'écrire la classe utilitaire **FileUtility.java** utilisée pour les actions sur les fichiers (découpage, recomposition, comptage de lignes, ...). Il a complété et finalisé les classes de cryptage et décryptage ; implémenté la classe générant une clef secrète à partir d'un mot de passe ; fait les tests en console avant d'implémenter l'interface ; complété la JavaDoc (relue et corrigée par Valériane). Il s'est amusé à ajouter quelques petits « plus » décrits plus loin.

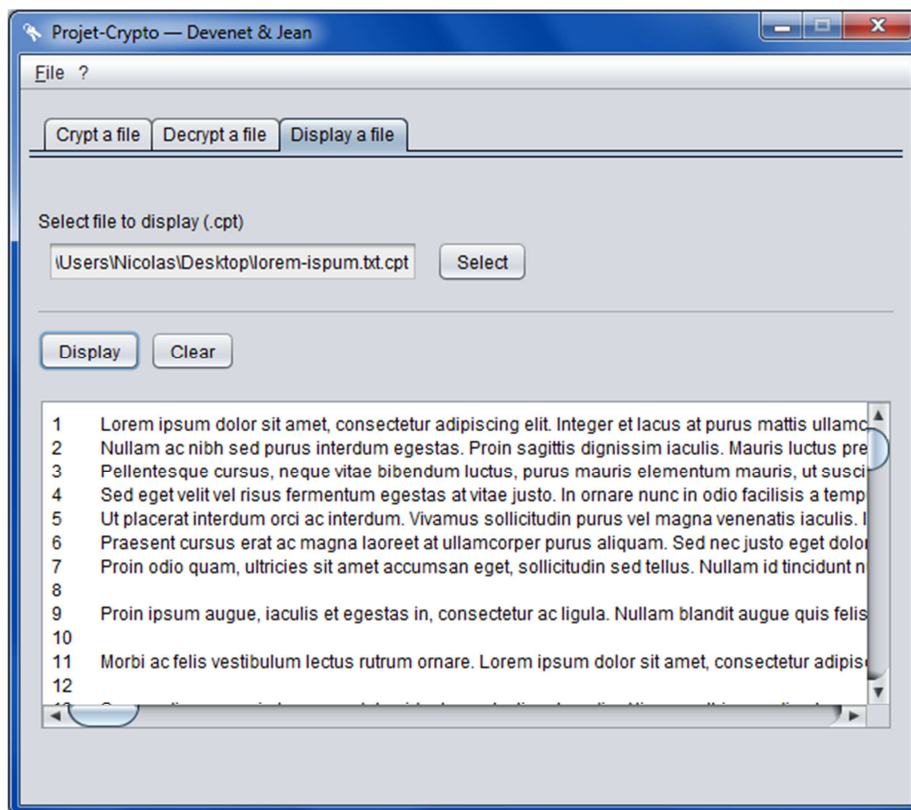
## ILLUSTRATIONS



*Onglet « Crypt a file »*



*Onglet « Decrypt a file »*



*Onglet « Display a file »*

## LES PETITS PLUS...


Pour améliorer l'interface et l'ergonomie de l'utilisateur, quelques petites fonctionnalités ont été mises en place.

- Menus accessibles directement avec des touches raccourcis (Alt + Q pour quitter, Alt + H pour afficher l'aide, ...).
- Ajout des fenêtres d'information « Help » et « About ».
- Les boutons principaux (*Crypt*, *Decrypt* et *Display*) ne sont cliquables que lorsque les champs précédents sont remplis.
- Désactivation de certains champs lorsque l'utilisateur n'a pas fait les étapes nécessaires.  
*Par exemple, dans l'onglet « Crypt a file », impossibilité de choisir les lignes à crypter si aucun fichier n'a été préalablement sélectionné ; numéro de fin de ligne automatiquement augmenté si le numéro de début de ligne est supérieur ; ...*
- Lors de l'affichage d'un fichier dans l'onglet « Display a file », les lignes sont numérotées.
- Ajout d'une icône pour l'application.
- Création d'un **Projet-Crypto.jar** pour lancer l'application sans devoir générer le projet.

Du côté développement, les « fichiers noyaux » sont totalement indépendants de l'interface graphique. On peut donc reprendre les packages **core** et **file** pour faire une librairie à utiliser facilement dans un autre projet, nécessitant le cryptage partiel de fichiers.

## SOURCES

Certaines méthodes ou implémentations utilisées sont directement inspirées de ressources trouvées sur Internet.

- Lire un fichier  
<http://tomtomgeek.blogspot.fr/2011/09/java-lire-un-fichier.html>
- Nombre de lignes dans un fichier  
<http://stackoverflow.com/questions/453018/number-of-lines-in-a-file-in-java>
- Lire la dernière ligne d'un fichier  
<http://fr.softuses.com/94739>
- Créer une fenêtre de sélection de fichiers/dossiers  
<http://www.zentut.com/java-swing/jfilechooser/>
-  Icône trousseau de clef  
<http://thenounproject.com/noun/key>