

Mini-Projet Java Card  
Applet « Identity »

2022/2023

ENSA FES

JAVA CARD

GSEII 2

MOHAMMED HANDA

ILYAS NHASSE

HAMZA LAKOUTI

Pr : M.FADAL

Introduction :

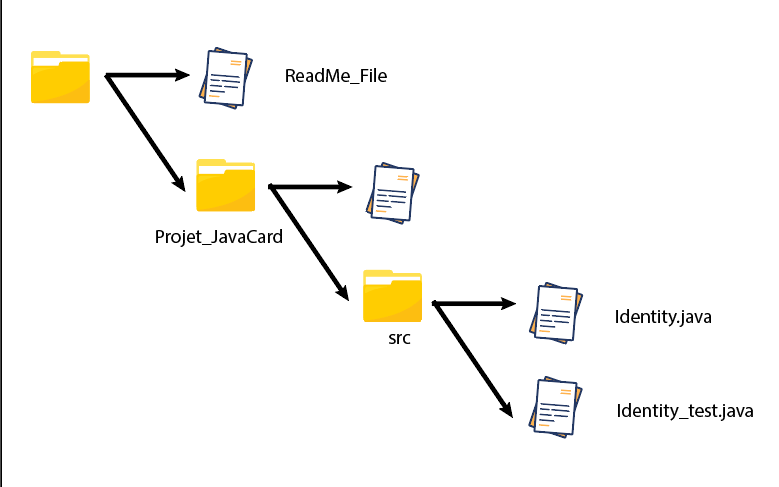
Objectif : Mettre en pratique les notions et connaissances acquises lors du cours et TP javacard, en réalisant un mini-projet de développement d’une applet d’identité élémentaire.

S’inspirer de l’exemple d’applet « Wallet » (déroulé lors du cours, et utilisé dans le TP)  
afin de mettre à jour l’applet « Identity » comme spécifié ci-dessous. La vérification et debugger l’applet « Identity » se fera à l’aide d’un fichier de tests déjà rédigé en java : « Identity\_test.java ».

Le Projet de fin de module de Java embarqué est constitué de deux fichier « Identity.java » et « Identity\_test.java » ce dernier test le bon fonctionnement l’ensemble des fonction développer dans l’autre fichier ; l’Applet à développer a quatre APDU chacune de ces APDU a son propre tache à faire ; alors on a 4 taches à manipuler :

* Ecriture
* Lecture
* Sauvegarde
* Modification

Structure des Fichiers :



Cette classe représente une applet de carte d'identité installée sur une carte à puce. L'applet dispose de plusieurs méthodes pour stocker, lire et mettre à jour les données stockées sur la carte.

La méthode d'installation est utilisée pour installer l'applet sur la carte à puce. Il crée une nouvelle instance de la classe Identity et l'enregistre.

La méthode de traitement est appelée lorsque la carte à puce reçoit une commande APDU (abréviation de Application Protocol Data Unit). Cette méthode détermine quelle commande est demandée en fonction de la valeur de l'octet INS (Instruction) dans l'en-tête APDU, puis appelle la méthode correspondante pour exécuter la commande.

La méthode storeData est utilisée pour stocker des données sur la carte. Il prend un objet APDU comme argument et vérifie que la commande est envoyée avec les paramètres corrects. Il stocke ensuite les données dans le tableau tableau\_identite.

La méthode readData est utilisée pour lire les données de la carte. Il prend un objet APDU comme argument et vérifie que la commande est envoyée avec les paramètres corrects. Il récupère ensuite les données demandées dans le tableau tableau\_identite et les renvoie à l'appareil demandeur.

La méthode updateDate permet de mettre à jour la date de naissance stockée sur la carte. Il prend un objet APDU comme argument et vérifie que la commande est envoyée avec les paramètres corrects. Il vérifie ensuite la clé MAC incluse dans la commande et, si la clé est valide, met à jour la date de naissance dans le tableau tableau\_identite.

La méthode updateID est utilisée pour mettre à jour l'ID stocké sur la carte. Il prend un objet APDU comme argument et vérifie que la commande est envoyée avec les paramètres corrects. Il déchiffre ensuite l'ID inclus dans la commande et met à jour l'ID dans le tableau tableau\_identite.

1/ private void storeData(APDU apdu);

Cette méthode stocke des données dans un tableau d'octets appelé tableau\_identite basé sur les valeurs des champs CLA et P1 dans la commande APDU (Application Protocol Data Unit). Le champ CLA identifie la classe d'instructions et le champ P1 identifie l'instruction au sein de cette classe.

La méthode vérifie d'abord si le champ CLA a la valeur correcte, et si ce n'est pas le cas, elle lève une exception avec le code d'erreur 0x6E00. Il vérifie ensuite si les champs P1 et P2 ont les valeurs correctes, et si ce n'est pas le cas, il lève une exception avec le code d'erreur 0x6B00.

Le procédé vérifie ensuite si le champ Lc (qui spécifie la longueur du champ de données dans la commande APDU) a la valeur correcte en fonction de la valeur du champ P2. Si le champ Lc a une valeur incorrecte, la méthode lève une exception avec le code d'erreur 0x6700.

De plus, les deux cases qu’on ajouter pour avoir la longueur de Nom et Prénom ont leur valeurs « tableau\_identite[Nom\_length\_index] = lc; » les deux valeurs vont etre utiliser dans la fonction readData() ;

Enfin, la méthode stocke les données entrantes dans le tableau tableau\_identite en fonction de la valeur du champ P2. Si le champ P2 a la valeur 0x01, il stocke les données sous la forme d'un nom commençant à la position Nom\_start\_index dans le tableau et remplit le premier des deux éléments de tableau ajoutés avec la longueur des données. Si le champ P2 a la valeur 0x02, il stocke les données en tant que prénom commençant à la position Prenom\_start\_index dans le tableau et remplit le deuxième des deux éléments de tableau ajoutés avec la longueur des données. Si le champ P2 a la valeur 0x03, il stocke les données en tant que date de naissance dans les positions jour\_de\_naissance\_index, mois\_de\_naissance\_index, annee\_de\_naissance\_MSB\_index et annee\_de\_naissance\_LSB\_index dans le tableau. Si le champ P2 a la valeur 0x04, il stocke les données en tant qu'ID commençant à la position numero\_identite\_start\_index dans le tableau.

2/ private void readData(APDU apdu) :

Cette méthode lit les données d'un tableau d'octets appelé tableau\_identite en fonction des valeurs des champs P1 et P2 de la commande APDU (Application Protocol Data Unit) et les renvoie à l'appelant APDU. Le champ P1 spécifie l'instruction dans la classe d'instructions et le champ P2 spécifie l'instruction dans cette instruction.

La méthode vérifie d'abord si le champ CLA (qui identifie la classe d'instructions) a la valeur correcte, et si ce n'est pas le cas, elle lève une exception avec le code d'erreur ISO7816.SW\_CLA\_NOT\_SUPPORTED. Il vérifie ensuite si les champs P1 et P2 ont les valeurs correctes, et si ce n'est pas le cas, il lève une exception avec le code d'erreur ISO7816.SW\_WRONG\_P1P2.

La méthode vérifie ensuite si le champ Le (qui spécifie le nombre maximal d'octets de données de réponse attendus) a la valeur correcte en fonction de la valeur du champ P2. Si le champ Le a une valeur incorrecte, la méthode lève une exception avec le code d'erreur 0x6C00 + Le.

Enfin, la méthode lit les données du tableau tableau\_identite en fonction de la valeur du champ P2 et les renvoie à l'appelant APDU. Si le champ P2 a la valeur 0x01 ou 0x02, il lit le nom et le prénom à partir des positions Nom\_start\_index et Prenom\_start\_index dans le tableau, respectivement, et les renvoie. Si le champ P2 a la valeur 0x03, il lit la date de naissance à partir des positions jour\_de\_naissance\_index, mois\_de\_naissance\_index, dnnee\_de\_naissance\_MSB\_index et annee\_de\_naissance\_LSB\_index dans le tableau et la renvoie. Si le champ P2 a la valeur 0x04, il lit l'ID à partir de la position numero\_identite\_start\_index dans le tableau et le renvoie.

3/ private void updateDate(APDU apdu):

Cette méthode est pour mettre à jour la date de naissance stockée dans un tableau d'octets appelé tableau\_identite. La méthode vérifie d'abord si le champ CLA (qui identifie la classe d'instructions) et les champs P1 et P2 (qui spécifient l'instruction dans la classe d'instructions) dans la commande APDU (Application Protocol Data Unit) ont les valeurs correctes. Si l'un de ces champs a une valeur incorrecte, la méthode lève une exception avec le code d'erreur correspondant.

La méthode vérifie également si le champ Lc (qui spécifie la longueur du champ de données dans la commande APDU) a la valeur correcte. Si le champ Lc a une valeur incorrecte, la méthode lève une exception avec le code d'erreur 0x6700.

Après ces vérifications, le procédé extrait la nouvelle date de naissance et la clé MAC (Message Authentication Code) de la commande APDU et vérifie que la clé MAC est correcte. Si la clé MAC est incorrecte, la méthode lève une exception avec le code d'erreur 0x6982.

Si toutes les vérifications réussissent, la méthode met à jour la date de naissance dans le tableau tableau\_identite en commençant à la position jour\_de\_naissance\_index.

4/ private void updateID(APDU apdu):

Cette méthode est pour mettre à jour l'ID stocké dans un tableau d'octets appelé tableau\_identite. La méthode vérifie d'abord si le champ CLA (qui identifie la classe d'instructions) et les champs P1 et P2 (qui spécifient l'instruction dans la classe d'instructions) dans la commande APDU (Application Protocol Data Unit) ont les valeurs correctes. Si l'un de ces champs a une valeur incorrecte, la méthode lève une exception avec le code d'erreur correspondant.

La méthode vérifie également si le champ Lc (qui spécifie la longueur du champ de données dans la commande APDU) a la valeur correcte. Si le champ Lc a une valeur incorrecte, la méthode lève une exception avec le code d'erreur 0x6700.

Après ces vérifications, la méthode extrait l'ID chiffré de la commande APDU et le déchiffre à l'aide de l'objet m\_decryptCipher. Il met ensuite à jour l'ID dans le tableau tableau\_identite en commençant à la position numero\_identite\_start\_index.

Conclusion :

Ce projet était surtout un exercice pédagogique qui nous permet de s’exercer sur plusieurs notion en Smart Card technologies, et d’avoir l’opportunité de développer des applets en Java Card.