Exercice en Java Card RMI
Exemple du porte-monnaie électronique

Le but de ce TP est de programmer l'exemple du porte-monnaie électronique en utilisant Java Card RMI.

Question 1 :
En vous inspirant du cours, écrire l'interface *WalletInterface.java* matérialisant les services offerts par l'objet distant.
Ensuite écrire la classe *Wallet.java* qui matérialise l'objet distant. Cette classe doit implémenter l'interface *WalletInterface.java* en complétant chaque méthode définie dans l'interface.
Une fois ces classes compilées, écrire la classe *Serveur.java* comme défini dans le cours. Après compilation, générer un fichier .CAP et installer l'applet Serveur.

Question 2 :
En vue de tester l'applet Serveur installée sur la carte, écrire le programme client comme indiqué en cours.
Compléter la classe *PCSCAccessor.java* pour permettre la connexion au lecteur comme tout client Java (incluant la JSR 268).

```
package rmiClient;

import java.io.IOException;

import javax.smartcardio.Card;
import javax.smartcardio.CardChannel;
import javax.smartcardio.CardTerminal;
import javax.smartcardio.CardTerminals;
import javax.smartcardio.CommandAPDU;
import javax.smartcardio.ResponseAPDU;
import javax.smartcardio.TerminalFactory;

import com.sun.javacard.clientlib.CardAccessor;

/*
 * commenter !! :D
 */

public class PCSCAccessor implements CardAccessor {
// a completer
	CommandAPDU capdu;
	ResponseAPDU rapdu;

	public PCSCAccessor(){
		// a completer
		}
		catch (Exception e) {
			e.printStackTrace();
			System.exit(1);
		}finally{
		}
	}

	public void closeCard() throws Exception {
```

```
		try{
			c.disconnect(true);
		}catch(javax.smartcardio.CardException c){
			System.out.println(c.getMessage());
		}catch(Exception e){
			e.printStackTrace();
		}
	}

	public byte[] exchangeAPDU(byte[] arg0) throws IOException {
		try{
			byte[] rapdu = cc.transmit(new CommandAPDU(arg0)).getBytes();


			//extraction de SW a la fin de rapdu
			byte[] sw = new byte[2];
			sw[0] = rapdu [rapdu.length-2];
			sw[1] = rapdu [rapdu.length-1];

			//mise du SW en tete de retour et copie du reste de rapdu.
			byte []retour = new byte [rapdu.length];
			retour[0]= sw[0];
			retour[1]= sw[1];
			for (int i = 0; i<rapdu.length-2; i++){
				retour[i+2] = rapdu[i];
			}
			return retour;
		}catch(javax.smartcardio.CardException ce){
			System.out.println(ce.getMessage());
			return null;
		}
	}
			/*
			 * elements pour débugger le protocole :
			System.out.print("<E>");
			System.out.println(Util.byteArrayToHexString(arg0, " "));
			String s = Util.byteArrayToHexString(rapdu, " ");
			ResponseAPDU r = new ResponseAPDU(rapdu);
			System.out.println("<Recu>\t\t"+s);
			System.out.println("<R>"+ Util.hexToASCII(s));
			System.out.println("<R>>"+
Util.hexToASCII(Util.byteArrayToHexString(r.getData(), " ")) +"\n"+ r.toString());

			s = Util.byteArrayToHexString(retour, " ");
			System.out.println("<Retour>\t"+s);*/


	public String toString(){
		return "card accessor PCSC gp [made in CEDRIC]";
	}

}
```