On va regarder chaque fichier et noter ce qu’il y a à changer dans le code pour faire fonctionner notre borne.

# **init.py**

Il faudra mettre la bb en mode caisse auto donc il faudra créer à la place du mode C pour caisse le mode CA pour caisse auto.

# **Loop.py**

C’est la que la boucle d’action principale est stocké

Exemple de boucle pour une caisse ou une caisse de Kve:

# Si la box est une caisse, on entre dans un contexte d'ajout d'argent

if setting.nomBox[0]=="C":

montant=MENU\_getMontant(argent) #Demande du montant à ajouter sur la carte

produit="RechargeMontant" #Le produit est nommé RechargeMontant(utiliser pour différentier les requêtes SQL)

nombre=1 #Une seule recharge (permet de standardiser les transactions mais est inutile ici)

reference=-1 #Pas de référence

# Si la box est une Kve, on entre dans un contexte de soustraction d'un montant libre

elif setting.nomBox[0]=="K":

montant=-MENU\_getMontant(argent)#Demande du montant à retirer sur la carte

produit="VenteMontant"#Le produit est nommé RechargeMontant(utiliser pour différentier les requêtes SQL)

nombre=1 #Une seule recharge (permet de standardiser les transactions mais est inutile ici)

reference=-1 #Pas de référence

#Les autres cas correspondent à des babass à un pianss

else:

reference,nombre,produit,montant=MENU\_getCommande(argent) #Paramètres de la commande

if montant==0: #Si la carte a été retirée

break #Arret de la transaction

newMontant=argent+montant #Calcul du nouveau montant de la carte

# Si le nouveau montant est négatif:

if newMontant<0:

hint("CREDIT INSUFFISANT",2)

hint("NE PAS SERVIR",3)

break #Arret de la transaction

# Si le nouveau montant n'est pas négatif, on effectue le débuquage sur la bdd puis sur la carte

DATA\_add(setting.projet\_path+'PICONFLEX2000-LOGS/LOG\_QUERRY.txt',QUERRY\_addArgent(STRING\_uidStrToInt(UID),montant)+QUERRY\_addTransaction(produit,nombre,setting.numeroBox,STRING\_uidStrToInt(UID),montant,reference)) #Ajout des requetes pour la BDD

hint("NE PAS RETIRER CARTE",4) #Avertissement sur lequel il faut lourdement insister en mode hors ligne!

RFID\_setArgent(newMontant,UID) #Ecriture du nouveau montant

## **Création des requêtes pour la BDD**

DATA\_add(setting.projet\_path+'PICONFLEX2000-LOGS/LOG\_QUERRY.txt',QUERRY\_addArgent(STRING\_uidStrToInt(UID),montant)+QUERRY\_addTransaction(produit,nombre,setting.numeroBox,STRING\_uidStrToInt(UID),montant,reference)) #Ajout des requetes pour la BDD

Il faut donc auparavant effectuer la récupération de l’argent depuis Lydia et repérer le montant voulu.

Il faut donc modifier la fonction MENU\_getMontant(argent) avec en argument argent l’argent déjà présent sur la carte.

MENU\_getMontant(argent) #Demande du montant à ajouter sur la carte

Définition de getMontant :

#Correction du 17/05 : Lorsque la carte est retiré, la fonction CLAVIER\_getRFID ne renvoi plus 0 mais -1.

#De même, la fonction CLAVIER\_getnoRFID renvoie -1 au lieu de 0

def MENU\_getMontant(argent):

montant=""

while True:

if (montant==""):

hint("ENTRER LE MONTANT",4)

montant = "0"

else:

hint("Montant: "+STRING\_montant(montant),4)

if setting.nomBox[0]=="C":

hint(STRING\_montant(argent)+" -> "+STRING\_montant(int(montant)+argent),2)

elif setting.nomBox[0]=="K":

hint(STRING\_montant(argent)+" -> "+STRING\_montant(int(montant)-argent),2)

\_touche=CLAVIER\_getRFID()

if (\_touche==None):#carte retiree

return 0

elif (\_touche==10):#\_touche entrer

if (montant==""):

pass

elif (int(montant)<config.minMontant):

hint("Montant Trop bas",4)

sleep(1)

elif ((int(montant)+argent>config.maxMontant) and ((setting.nomBox[0]=="C"))):

hint("Total Trop haut",4)

sleep(1)

elif ((int(montant)>config.maxTransaction) and ((setting.nomBox[0]=="C"))):

hint("Montant Trop haut",4)

sleep(1)

else:

return int(montant)

elif (\_touche in [46,127]):#touches del/backspace

if (len(montant)!=0):

montant=montant[0:-1]

else:

pass

elif(\_touche in [48,49,50,51,52,53,54,55,56,57]):#touches 0,1,2,3,4,5,6,7,8,9

montant=str(montant)+chr(\_touche)

elif (\_touche==43):#\_touche +

if (montant==""):

montant=str(500)

elif (int(montant)<500):

montant=str(500)

elif (int(montant)<1000):

montant=str(1000)

elif (int(montant)<2000):

montant=str(2000)

elif (int(montant)<5000):

montant=str(5000)

elif (int(montant)<10000):

montant=str(10000)

elif (int(montant)<20000):

montant=str(20000)

elif (int(montant)<50000):

montant=str(50000)

elif (\_touche==45):#\_touche -

if (montant==""):

montant=""

elif (int(montant)>50000):

montant=str(50000)

elif (int(montant)>20000):

montant=str(20000)

elif (int(montant)>10000):

montant=str(10000)

elif (int(montant)>5000):

montant=str(5000)

elif (int(montant)>2000):

montant=str(2000)

elif (int(montant)>1000):

montant=str(1000)

elif (int(montant)>500):

montant=str(500)

elif (\_touche==42):#\_touche \*

REZAL\_reboot()

elif (\_touche==47):#\_touche /

return REZAL\_restart()

elif (\_touche in [0,9]): #\_touche TAB ou Calculator

error = MENU\_getCode(config.codeHelper, "Code Helper", [0, 9], True) # On demande le code Helper et on donne la touche "calulator" ou la touche tab comme exitKeys et en vérifiant la présence d'une carte

if error != -1:

MENU\_rapportPbCarte()

C’est cette fonction qu’il va falloir copier et modifier pour qu’elle accepte les paiements lydia.

On va modifier cette fonction en une fonction:

def MENU\_getMontantLydia(argent):

montant=""

while True:

if (montant==""):

hint("Présenter le QrCode",4)

montant = "0"

else:

hint("Montant: "+STRING\_montant(montant),4)

if setting.nomBox[0]=="C":

hint(STRING\_montant(argent)+" -> "+STRING\_montant(int(montant)+argent),2)

elif setting.nomBox[0]=="K":

hint(STRING\_montant(argent)+" -> "+STRING\_montant(int(montant)-argent),2)

\_touche=CLAVIER\_getRFID()

elif setting.nomBox[0]=="A":

montant=MENU\_getQrCode()# ici on récupère le montant donné par le Client avec son QRCode et cela après vérification du montant ( ni trop gros ni trop faible)

hint(STRING\_montant(argent)+" -> "+(int(montant)+argent,2))

On s’aidera de la fonction suivante pour récupérer les informations du QrCode.

def MENU\_getQrCode():

#A écrire la fonction doit récupérer un QRcode et donner le montant et les autres infos liées au QRcode

return (montant, autre infos ... )

Le code pour les caisse automatique sera “A” comme Automatique par analogie avec “C” des caisses

RFID.py

def RFID\_readCarte():

if config.debugging:

print("## RFID\_readCarteTEST ##")

while True:

try:

if RFID\_presence():

(status,uid)=MIFAREReader.MFRC522\_SelectTagSN()

uidstring=STRING\_Tag(uid,len(uid))

if config.debugging:

print("uid:"+uidstring)

if status==MIFAREReader.MI\_OK:

argentListe = RFID\_readblock(config.blockArgent,uidstring)

hashGuincheListe = RFID\_readblock(config.blockHashCodeGuinche,uidstring)

hashUIDListe = RFID\_readblock(config.blockHashUID,uidstring)

hashArgentListe = RFID\_readblock(config.blockHashArgent,uidstring)

return (uidstring,int(STRING\_Tag(argentListe),16),STRING\_Tag(hashGuincheListe),STRING\_Tag(hashUIDListe),STRING\_Tag(hashArgentListe))

elif config.debugging:

print("absence de carte")

except:

hint("! PROBLEME LECTURE !",4)

sleep(1)

hint("",4)

if config.debugging:

#permet d'essayer une seule fois par seconde de lire une carte en mode debug, pour ne pas remplir la console

sleep(1)

sleep(0.05)

Le code pour lire la carte avec le lecteur RFID ca régale.