Systèmes embarqués

Travaux Pratiques

Transfert de données sur Raspberry Pi

Julien Marot

1 Introduction, python sur Raspberry

1.1 Introduction

Vous disposez d'un PC classique et d'un setup Raspberry Pi. L'objectif du TP est de générer des séries numériques avec le logiciel Python, d'une part sur Raspberry Pi, d'autre part sur PC. Le langage Python est du type 'texte structuré'. Il est similaire à Matlab car il inclut des boucles, des tests, et des fonctions. Il est aussi similaire au langage C car on importe des bibliothèques de fonctions (de même que l'on inclut des fichiers .h en c). Il ressemble aussi au C++ car il permet de faire de la programmation orientée objet. Dans le cadre de ce TP, nous souhaitons créer des séries numériques à l'aide de fonctions Python, et lancer ces programmes Python sur une Raspberry Pi. Nous effectuons donc des travaux similaires à ceux réalisés sous Matlab lors d'autres TPs, avec du matériel et un logiciel beaucoup moins chers puisque le Raspberry Pi vaut 35 euros et Python est un logiciel libre gratuit.

1.2 Setup hardware

Vous disposez pour chaque binôme d'un setup Raspberry Pi complet. C'est en dernier que l'on alimente le RPi !
Branchez le RPi à un écran via un adaptateur HDMI-VGA et un câble VGA. Branchez une souris, un clavier, et un câble Ethernet. Branchez le câble Ethernet au réseau. Enfin, branchez l'alimentation.
Question 1 : Si vous ne démarrez pas en mode 'Interface graphique', sur la ligne de commande tapez `sudo startx`. Cela est inutile si vous utilisez un RPi 3. Savez-vous quel modèle vous utilisez ? Est-il plus rapide ou plus lent qu'un Pi 3 à votre avis ? Utilisez, pour le

savoir, les commandes suivantes :
cat /proc/device-tree/model
mais aussi, pour des informations complémentaires :
pinout
cat /proc/cpuinfo

Question 2 : Lancez l'interface Python.
Pour cela allez dans Menu>Programming. Combien de versions de Python sont installées ? Choisissez Python 2. On peut aussi utiliser python sur la ligne de commande, en faisant précéder le nom d'un programme de 'python'. Créez un dossier sur le bureau à votre nom. Dedans, enregistrez votre nouveau fichier sous le nom `Fonctions.py` et complétez-le avec

```
print("bonjour")
```

Question 3 : Exécutez votre programme : à partir du fichier faites Run>Run Module. Voyez-vous `bonjour` apparaître sur la ligne de commande ? Lancez ce même programme à partir de la ligne de commande : faites `python Fonctions.py`. Le "bonjour" apparaît-il ?

Créez un nouveau fichier nommé `SeriesNumeriques.py`.

1.3 Exemples de fonctions et de programmes Python

Vous devez maintenant compléter vos deux fichiers : celui contenant les fonctions et celui permettant de tester ces fonctions en créant des séries numériques. Voici, dans les algorithmes 1 et 2 ci-dessous, des exemples de fonctions et un exemple d'utilisation de ces fonctions :

Algorithm 1 Fichier de définition *Fonctions.py*

```
def add(a,b):
c=a+b
return c

def mult(a,b):
c=a*b
return c

def testnombre(a,seuil):
if a<seuil:
print("petit")
else:
print("grand")

def boucle(indice):
n=0;
while n<=indice:
print(n)
n=n+1

print("boucle finie")
```

Question 4 : Complétez vos fichiers avec les exemples qui vous sont donnés. Respectez bien la syntaxe (les ' : ' et les indent notamment). Parvenez-vous à exécuter le programme `SeriesNumeriques.py` ? Qu'affiche-t-il en résultat d'exécution ?

1.4 Création de fonctions et de programmes Python

Vous allez maintenant créer vos propres fonctions.

Question 5 : Complétez le fichier `Fonctions.py` avec les fonctions `power(a,n)` qui calcule a^n et `factorielle(n)` qui calcule `n!`. Notez pour cela que `n!=n*(n-1)!`.

Testez ces fonctions en complétant le programme `SeriesNumeriques.py` : affichez le résultat à chaque fois avec la fonction `print`.

Algorithm 2 Fichier de test *SeriesNumeriques.py*

```
import math
from Fonctions import add, mult, testnombre, boucle

a=2
b=3
testnombre(a,b)
boucle(b)

c=add(a,b)
print(c)

d=mult(a,b)
print(d)
```

Question 6 : Implantez dans le programme `SeriesNumeriques.py` une boucle for affichant toutes les valeurs de n! de 1 à une valeur souhaitée.

1.5 Transfert de fichier et test sur PC

Nous souhaitons maintenant tester les programmes sur un PC classique. Pour cela il faut y transférer les programmes. Allumez un PC, bootez sous Ubuntu. Si vous êtes nombreux, plusieurs binômes l'auront déjà fait. Vous allez coopérer avec eux pour étudier la façon dont leur programmes fonctionnent.
Question 7 : Sur votre Raspberry Pi : placez vos programmes Python `Fonctions.py` et `SeriesNumeriques.py` dans le dossier /home/pi. Retrouvez-vous le fichier `Client_Web.py` dans ce même dossier ou sur le Desktop ? A quoi sert-il ? On peut aussi télécharger `Client_Web.py` sur le site suivant :
`http://www.fresnel.fr/perso/marot/#Teaching`
Le fichier `Client_Web.py` peut aussi être lancé sur un PC démarré sous Ubuntu.

Question 8 : Vous pouvez, grâce à l'application `Client_Web.py`, autoriser d'autres ordinateurs à accéder à votre Raspberry Pi (ou à votre ordinateur). Lancez l'application `Client_Web.py`, à l'aide de la commande suivante sur un terminal :
`Python Client_Web.py`.
Ouvrez une fenêtre Firefox sur le PC (au préalable, vérifiez que son câble Ethernet est bien branché au réseau). Si vous disposez d'un Raspberry Pi 3 B ou B plus, il est muni du WiFi et vous pouvez faire un partage WiFi via votre téléphone.
Vous pouvez retrouver l'adresse IP de votre raspberry en tapant `hostname -I` ou `ifconfig` sur votre ligne de commande. Pour le WiFi, vous trouver l'adresse IP de votre Raspberry au niveau de 'wlan0'.
Ecrivez `172.17.107.15x:8000` sur la barre d'adresse Firefox, où `172.17.107.15x` est l'adresse IP de votre Raspberry Pi (ce n'est qu'un exemple). Téléchargez les programmes que vous avez écrits sur votre Raspberry Pi (clic droit > Enregistrer sous). Les deux programmes `Fonctions.py` et `SeriesNumeriques.py` sont-ils présents sur le Bureau ?

Question 9 : Sur le PC, sur un terminal, faites `cd Bureau`. De façon similaire à ce qui est montré en figure **??** pour un programme simple 'hello.py', exécutez le fichier
`SeriesNumeriques.py` : `python SeriesNumeriques.py`.
Les résultats obtenus sont-ils les mêmes que sur la Raspberry ?

2 Tranfert et contrôle à distance

2.1 Instructions de transfert

Les instructions suivantes permettent d'envoyer un fichier sur un RPi distant. Testez-les :
`scp MONFICHIER pi@IP:Desktop` : j'envoie mon fichier vers un RPi.
`scp pi@IP:MONFICHIER` : je récupère un fichier depuis un RPi.

`rsync` permet de faire un transfert similaire, en sélectionnant tous les fichiers d'un type donné.

Question 10 : exécutez les différentes instructions de transfert.

2.2 Instructions de contrôle à distance

`ssh` permet de prendre le contrôle à distance.
Par exemple :
`ssh pi@192.168.0.19`.

Prenez le contrôle du RPi d'un collègue, transférez des fichiers !
Question 11 : Effectuez une double attaque : prenez le contrôle d'un RPi à partir duquel vous allez prendre le contrôkle d'un autre RPi ; prenez une photo, et transférez-là vers votre propre RPi avec l'une des instructions étudiées ci-dessus.

2.3 Contrôle à distance par WiFi

Question 12 : Parvenez-vous à prendre le contrôle à distance d'un RPi *via* WiFi ?

3 Conclusion

Débranchez votre setup Raspberry en commençant par l'alimentation. Rangez les cartes micro-SD, et le matériel. Refaites les branchements des ordinateurs aux périphériques que vous avez utilisés pour vos Raspberry Pi.