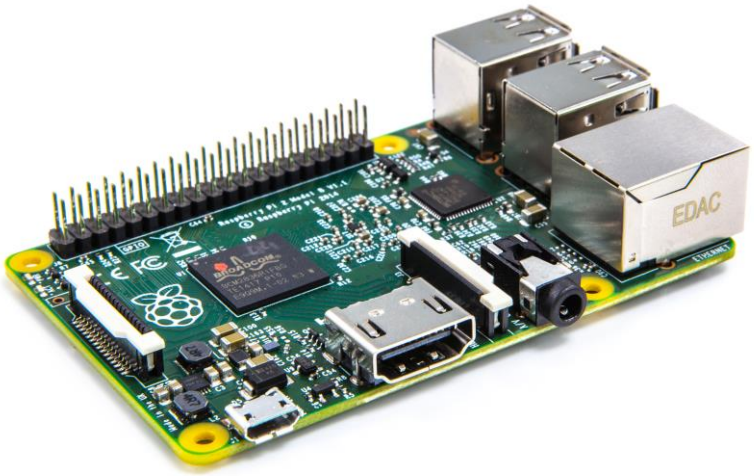
ISN

ISN – Informatique et Sciences du Numérique

DOCUMENT RESSOURCE : CARTE RASPBERRY PI 2



SOMMAIRE

SOMMAIRE **2**

1 – PRESENTATION **3**

2 – DIFFERENTS ELEMENTS DE LA CARTE RASPBERRY PI 2 ..... **4**
2.1 – CARTE RASPBERRY PI 2 4
2.2 – MICROPROCESSEUR ET PROCESSEUR GRAPHIQUE 4
2.3 – MEMOIRE 5
2.4 – CONNECTEURS DE LA CARTE RASPBERRY PI 2 5

3 – PORT GPIO **8**

4 – DEMARRAGE INITIAL **9**

5 – COMMANDES LINUX **10**
5.1 – ARBORESCENCE DU SYSTEME DE FICHIERS 10
5.2 – DROITS UTILISATEUR ET COMMANDE « SUDO » 11
5.3 – COMMANDES LINUX 11
5.3.1 – Manipulation de fichiers et de répertoires 11
5.3.2 – Exécution de programmes 12
5.3.3 – Installer des logiciels (ou paquets) 12
5.3.4 – Eteindre la carte Raspberry 13

6 – INTERFACE GRAPHIQUE **14**

7 – CONTROLE A DISTANCE DU RASPBERRY PAR UNE LIAISON SSH **15**

8 – LANGAGE PYTHON **19**
8.1 – PRESENTATION DU LANGAGE PYTHON 19
8.2 – CREATION D'UN FICHIER EXECUTABLE 20
8.3 – BIBLIOTHEQUE RPI.GPIO 21
8.3.1 – Fonction GPIO.setmode() 21
8.3.2 – Fonction GPIO.setup() : 21
8.3.3 – Fonction GPIO.output() 22
8.3.4 – Fonction GPIO.input() 22
8.3.5 – Fonction GPIO.add_event_detect() 23
8.3.6 – Fonction GPIO.remove_event_detect() 23
8.3.7 – Fonction GPIO.event_detect() 23
8.3.8 – Fonction GPIO.cleanup() 23

1 – PRESENTATION

La carte **Raspberry Pi 2** est une **carte électronique** qui mesure **56 mm x 85 mm**. Il s'agit de l'une des dernières versions des cartes Raspberry.

Elle est capable de faire fonctionner un **système d'exploitation Linux complet**. Il s'agit donc d'un ordinateur bon marché mais avec des fonctionnalités suffisamment puissantes. La carte Raspberry Pi, est essentiellement destinée à des applications d'informatique embarquée.

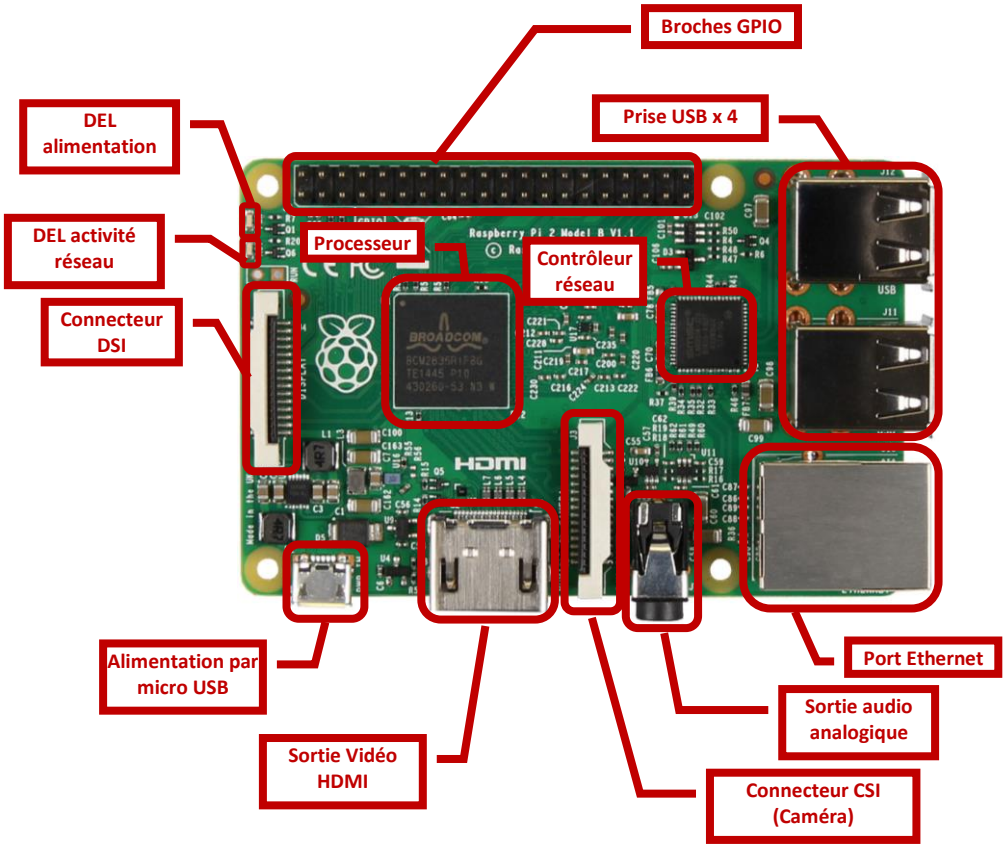
Le cœur de la carte est une **puce de type SoC** (System on a Chip). La mémoire, le microprocesseur et le processeur graphique sont physiquement placé sur cette même puce, ce qui permet de réduire la taille du circuit.

La Raspberry Pi a été conçu dès le départ pour faire fonctionner un **système d'exploitation Linux**. Il existe déjà plusieurs distributions de Linux utilisables avec le circuit BCM2836 du Raspberry Pi. Citons notamment Raspbian (Debian Raspberry Pi), Fedora Remix et Arch Linux.

La **version IoT** (Internet of the Thing) de **Windows 10** peut également être installée sur la carte rapsberry PI 2.

2 – DIFFERENTS ELEMENTS DE LA CARTE RASPBERRY PI 2

2.1 – CARTE RASPBERRY PI 2



2.2 – MICROPROCESSEUR ET PROCESSEUR GRAPHIQUE

Le SOC BCM2836 de Broadcom est équipé d'un **microprocesseur** quadruple coeur **ARMV7 Cortex A7** cadencé à **900 MHz**.

La carte Raspberry étant conçue pour des applications embraquée, le microprocesseur doit avoir une faible consommation. Une vitesse d'horloge relativement lente permet **d'abaisser la consommation d'énergie** et permet également un **fonctionnement sous une tension faible** ce qui induit une production de chaleur réduite et une durée de vie du microprocesseur plus longue.

Le processeur graphique est un **processeur VideoCore IV** double core. Il assure tous les **traitements vidéo et audio** demandés au SoC. Il prend en charge la norme OpenGL ES 2.0 qui permet **l'affichage 3D** pour systèmes embarqués. Ce processeur implémente de façon matérielle un **codec H.264**, 1080 p, 30 trames/sec indispensable pour la haute définition.

2.3 – MEMOIRE

Deux types de mémoire sont employés dans la carte Raspberry : la **DRAM** (**D**ynamic **R**andom **A**cess **M**emory) et la **flash micro-SD** (**S**ecure **D**igital).

La carte Raspberry PI 2 est dotée de **1 Go de mémoire DRAM** intégrée directement dans la puce SoC. Il s'agit de mémoire faible consommation conçue pour les applications mobiles.

La **carte mémoire flash micro-SD** permet de stocker le système d'exploitation, tous les programmes et les données qui doivent persister. La carte Raspberry utilise la carte mémoire SD de la manière qu'un ordinateur utilise un disque dur pour **conserver de façon permanente les données et les programmes** notamment le **système d'exploitation**.

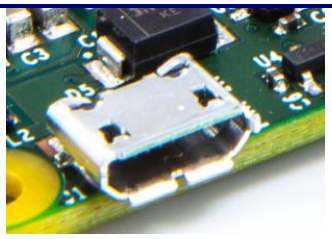
- Slot SD carte raspberry PI 2 -

2.4 – CONNECTEURS DE LA CARTE RASPBERRY PI 2

La carte Raspberry PI 2 est équipée de **11 connecteurs** : micro USB (Alimentation), HDMI (vidéo), connecteur jack 3,5 mm (audio analogique), connecteur RJ45 (Ethernet), connecteur GPIO (entrée/sorties numériques), connecteur CSI (caméra), connecteur DSI (afficheur) et quatre ports USB.

Connecteur micro USB d'alimentation

La carte Raspberry PI 2 est alimentée via le connecteur micro USB d'alimentation au moyen d'un **adaptateur secteur 5V -2 A**. En fonctionnement normal, la consommation de la carte Raspberry est d'environ **600 mA**.

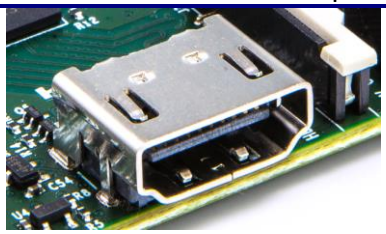


- Connecteur micro USB -

- Adaptateur secteur -

Sortie Vidéo Numérique (HDMI)

La carte Raspberry est capable de **générer des signaux vidéo et audio à la norme HDMI**. Il dispose pour cela d'un connecteur HDMI. Dans le cas où l'écran utilisé ne dispose pas de connecteur HDMI, il est nécessaire d'utiliser un adaptateur HDMI/VGA ou HDMI/DVI.



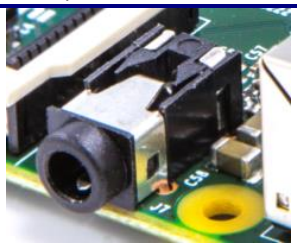
- Connecteur HDMI -

- Câble HDMI -

- Adaptateur HDMI/VGA -

Sortie audio analogique (jack stéréo 3,5 mm)

La carte Raspberry est capable de **générer une sortie audio stéréo**. Elle est disponible sur une prise jack stéréo standard de 3,5 mm.

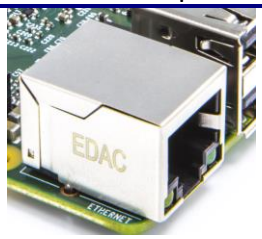


- Prise jack 3,5 mm -

- Câble audio stéréo 3,5 mm -

Connecteur RJ45

Le **connecteur RJ45** permet la connexion d'un câble réseau ou d'un dongle WiFi.



- Connecteur RJ45 -

- Câble Réseau -

- Clé WiFi -

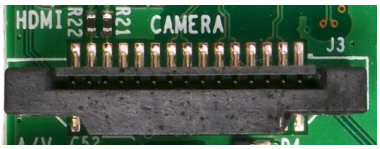
Connecteur GPIO

Le connecteur **GPIO** (**G**eneral **P**urpose **I**nput **O**utput) comprend **40 broches** réparties sur **2 rangées**. Ce connecteur supporte les **Entrées/Sorties numériques** mais également les **sorties PWM**, les **Entrées/Sorties de communication** et les **alimentations 5 V et 3,3 V**.

- Connecteur GPIO -

Connecteur CSI

Le connecteur **CSI** (**C**amera **S**erial **I**nterface) permet la connexion d'une **caméra ou d'un appareil photo** numérique via une **nappe souple à 15 conducteurs**.

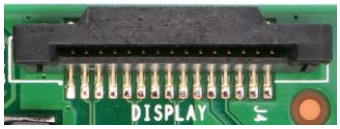


- Connecteur CSI -

- Module Caméra pour Raspberry -

Connecteur DSI

Le connecteur **DSI** (**D**isplay **S**erial **I**nterface) permet la connexion d'un **afficheur** via une **nappe souple à 15 conducteurs**.



- Connecteur DSI -

- Ecran tactile pour Raspberry -

Connecteurs USB

La carte Raspberry PI 2 est équipée de **4 ports USB**. Les périphériques USB connectés sur ces ports sont **automatiquement reconnus**.

Cependant, la carte Raspberry ne pourra alimenter que les périphériques « basse consommation » (souris, clavier, clés USB). Pour les autres périphériques qui ne possèdent pas leur propre alimentation il est conseillé d'utiliser un **concentrateur USB** avec alimentation.

- Connecteurs USB -

- Concentrateur USB -

3 – PORT GPIO

Le connecteur **GPIO** (**G**eneral **P**urpose **I**nput **O**utput) comprend **40 broches** réparties sur **2 rangées**. Il comporte **29 E/S numériques**, une interface **I2C**, une interface **SPI** et une interface **UART.**

Les broches du GPIO sont repérées soit par le numéro du connecteur GPIO (Board) ou bien par le numéro de la ligne d'E/S au niveau du µP (BCM).

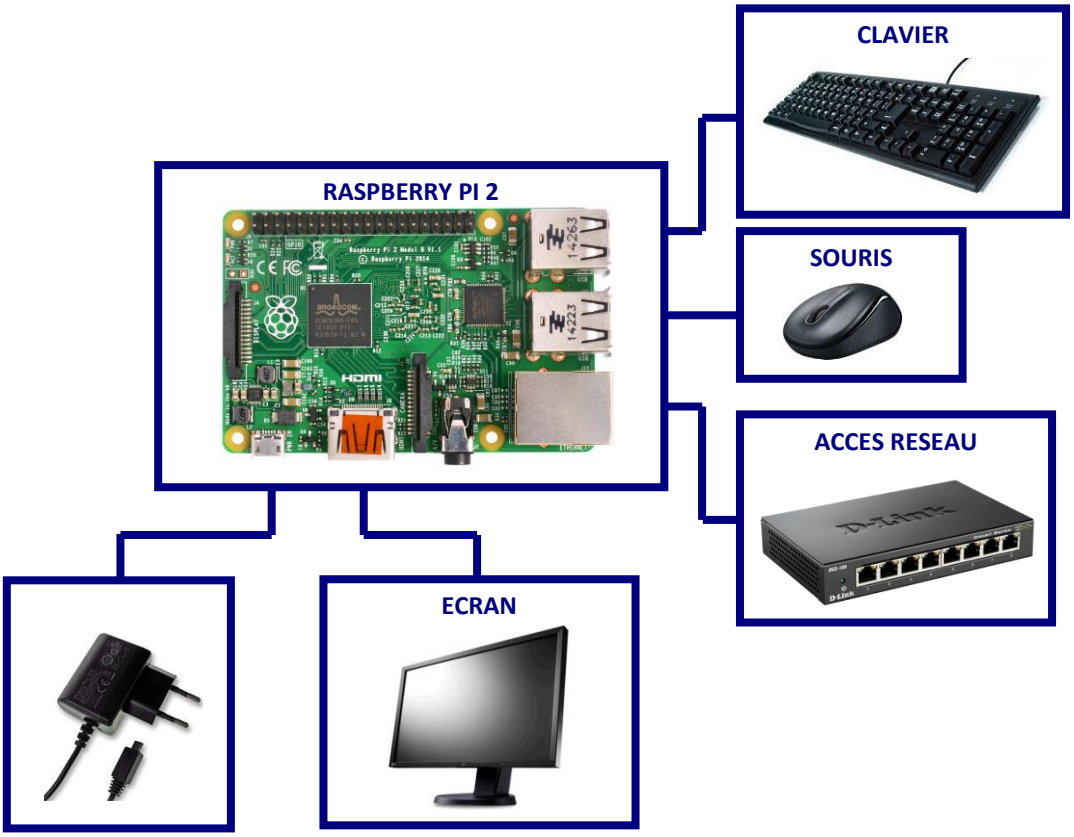
Fonction	N° BCM	N° Board
3V3		1
GPIO2 – SDA1	2	3
GPIO3 – SCL1	3	5
GPIO4 – GPCLK0	4	7
GND		9
GPIO17	17	11
GPIO27	27	13
GPIO22	22	15
3V3		17
GPIO10 – MOSI	10	19
GPIO9 – MISO	9	21
GPIO11 – SCLK	11	23
GND		25
SDA0		27
GPIO5 – GPCLK1	5	29
GPIO6 – GPCLK2	6	31
GPIO13 – PWM1	13	33
GPIO19 – PCM_FS	19	35
GPIO26	26	37
GND		39

N° Board	N°BCM	Fonction(s)
2		5V
4		5V
6		GND
8	14	GPIO14 – TXD
10	15	GPIO15 – RXD
12	18	GPIO18 – PCM_CLK
14		GND
16	23	GPIO23
18	24	GPIO24
20		GND
22	25	GPIO25
24	8	GPIO8 – CE0
26	7	GPIO7 – CE1
28		SCL0
30		GND
32	12	GPIO12– PCM0
34		GND
36	16	GPIO16
38	20	GPIO20 – PCM_DIN
40	21	GPIO21 – PCM_DOUT

Toutes les broches du connecteur GPIO opèrent à 3,3 V. La carte Raspberry ne dispose d'aucune protection contre les surtensions. Si par mégarde une tension de 5 V est appliquée sur une entrée GPIO, le RaspBerry sera détruit.

4 – DEMARRAGE INITIAL

La configuration minimale pour faire fonctionner la carte Raspberry est la suivante :



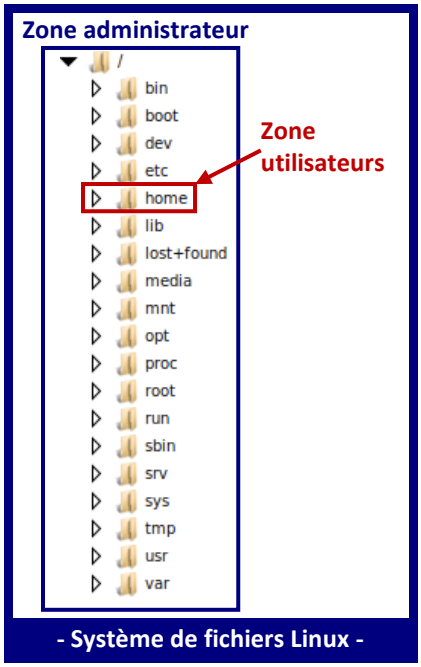
Attention à brancher l'alimentation de la carte Raspberry après avoir connecté tous les autres composants sinon ils ne seront pas reconnus lors de la phase d'initialisation et le système risque de ne pas fonctionner correctement.

5 – COMMANDES LINUX

5.1 – ARBORESCENCE DU SYSTEME DE FICHIERS

Lorsque l'on installe un système Gnu/Linux tel que Raspbian, l'utilitaire d'installation crée automatiquement une arborescence « type » qui est commune à la plupart des systèmes Gnu/Linux, à quelques variantes près. Cette arborescence est composée d'un ensemble de répertoires ayant des fonctions précises pour la gestion du système et qui contiennent selon les cas des fichiers exécutables, des fichiers de paramétrages ou de configuration.

La racine du système de fichier d'un système Gnu/Linux est symbolisée le symbole « **/** ». Ainsi, le répertoire / contient tous les autres répertoires du système.



- Système de fichiers Linux -

Répertoire	Contenu
/bin	**bin**ary utilities : programmes ou commandes (exécutables) susceptibles d'être utilisés par tous les utilisateurs de la machine
/boot	**boot**strap : Fichiers permettant à Linux de démarrer
/dev	**dev**ice : Fichiers et points d'entrée des périphériques
/etc	**e**diting **t**ext **c**onfig : Fichiers de configuration
/home	**home** directory : Répertoires et fichiers personnels des utilisateurs
/lib	**lib**rairies : Bibliothèques partagées et utilisées par les programmes
/lost+found	Données sans répertoire, perdues à cause d'un dysfonctionnement
/media	Points de montages pour les médias amovibles (disques durs externes, clés USB…)
/mnt	**mount** : Point de montage pour monter temporairement un système de fichiers
/opt	**opt**ional : paquets d'applications supplémentaires (optionnelles)
/proc	**proc**esses : répertoire virtuel pour les informations système
/root	répertoire de l'administrateur ou super-utilisateur
/run	**run**time system : Informations relatives au système depuis son dernier démarrage
/sbin	**s**uper **bin**nary utilities : Exécutables système réservés à l'administrateur
/srv	**serv**ices : Données pour les services du système
/sys	**sys**tem : répertoire virtuel pour les informations système
/tmp	**tem**porary : Fichiers temporaires des applications
/usr	**u**nix **s**ystem **r**esources : Hiérarchie secondaire, pour des données en lecture seule par les utilisateurs (applications usuelles des utilisateurs et leurs fichiers).
/var	**var**iable : Données variables et diverses

5.2 – DROITS UTILISATEUR ET COMMANDE « SUDO »

Le système Gnu/Linux est basé sur un système très puissant de **droits d'utilisateurs** : chaque utilisateur a des droits précis en termes de lecture, écriture ou exécution d'un programme, d'un fichier. Ce système de droits et de comptes utilisateurs sécurise les systèmes Gnu/Linux.

Un utilisateur n'a tous les droits (exécuter, lire, écrire) que dans une « zone » du système de fichier qui lui est réservée. En dehors de cette zone, il n'a pas les droits en exécution et écriture. L'utilisateur qui a « tous les droits » et qui peut faire ce qu'il veut et où il veut (effacer, exécuter, modifier, etc..) s'appelle l'**administrateur** ou « **root** » : cet utilisateur est créé par défaut.

La commande « sudo »permet d'obtenir les « droits root » pendant une durée limitée et pour une commande précise. Pour cela il faut faire précéder la commande à exécuter de la commande « sudo » :

```
$ sudo nomcommande -options params
```

5.3 – COMMANDES LINUX

5.3.1 – Manipulation de fichiers et de répertoires

Commande	Description
pwd	La commande « `pwd` » permet d'afficher le nom du répertoire courant.
cd	La commande « **`cd nom_répertoire`** » permet de se déplacer dans l'arborescence de fichiers : – **`cd /`** : permet de se placer à la racine. – **`cd /home`** : permet d'aller dans le répertoire « /home ». – **`cd ..`** : permet de remonter dans l'arborescence.
ls	La commande « **`ls`** » permet de **connaître le contenu** d'un répertoire : – **`ls`** : permet d'afficher le contenu du répertoire courant. – **`ls -a`** : permet d'afficher tous les fichiers, fichiers cachés compris. – **`ls -l`** : permet d'afficher la liste des fichiers avec les détails.
cp	La commande « **`cp source destination`**» permet de **copier un fichier ou un répertoire** : – **`cp home/prog1.py /home/Prog_Python/prog1.py`** : permet de copier le fichier « prog1.py » contenu dans le répertoire « /home », dans le répertoire « /home/Prog_Python ». – **`cp -r home/Prog_Python /home/Test`** : permet de copier le répertoire « Prog_Python » dans le répertoire « /home/Test ».
mkdir	La commande « **`mkdir nom_répertoire`** » permet de **créer un répertoire** : – **`mkdir build`** : permet créer le répertoire « build » dans le répertoire courant. – **`mkdir home/Prog_Python`** : permet de créer le répertoire « Prog_Python » dans le répertoire « home ».
rm	La commande « **`rm nom`**» permet de **supprimer un fichier ou un répertoire** : – **`rm home/Prog_Python/prog1.py`** : permet de supprimer le fichier « prog1.py » contenu dans le répertoire « /home/Prog_Python ». – **`rm -r home/Prog_Python`** : permet de supprimer le répertoire « Prog_Python » contenu dans le dossier « /home ».

5.3.2 – Exécution de programmes

Commande	Description
`chmod`	La commande « **`chmod +x nom_fichier`** » permet de se rendre un fichier exécutable : – **`chmod +x prog1.py`** : permet de rendre le programme prog1.py exécutable.
`python`	Pour exécuter un programme python il faut taper la commande « **python nom_fichier.py** » : – **`python prog1.py`** : permet d'exécuter le programme « prog1.py ».

5.3.3 – Installation des logiciels (ou paquets)

La distribution Raspbian est un ensemble d'éléments logiciels qui permettent de faire fonctionner la carte Raspberry et les programmes. Cette distribution est en fait l'assemblage de « briques » élémentaires appelées « **paquets** ». Il s'agit d'éléments de code source qui ont déjà été compilés par les mainteneurs de la distribution et qui sont « prêts à l'emploi » pour l'utilisateur final. Ces paquets sont stockés sur des serveurs dédiés accessibles gratuitement et librement à partir d'internet.

Au sein de système d'exploitation Raspbian, les paquets sont gérés à l'aide du **gestionnaire de paquet**. Ce programme, appelé « **apt** », permet d'installer de nouveaux paquets, de vérifier et d'installer les dépendances (liste de tous les autres paquets utiles à l'utilisation du nouveau paquet), mettre à jour la liste de paquets installés, etc...

Commande	Description
`apt-get install`	La commande « **`apt-get install nom_paquet`** » permet d'installer un nouveau paquet : – **`apt-get install geany`** : permet d'installer le logiciel « geany ».
`apt-get remove`	La commande « **`apt-get remove nom_paquet`** » permet de désinstaller un paquet sans effacer les fichiers de configuration : – **`apt-get remove geany`** : permet de désinstaller le logiciel « geany ».
`apt-get purge`	La commande « **`apt-get purge nom_paquet`** » permet de désinstaller un paquet ainsi que les fichiers de configuration : – **`apt-get purge geany`** : permet de désinstaller le logiciel « geany ».
`apt-get autoremove`	La commande « **`apt-get autoremove`** » permet de désinstaller les paquets inutiles.
`dpkg -l`	La commande « **`dpkg -l`** » permet d'obtenir la liste des paquets installés.
`apt-get update`	La commande « **`apt-get update`** » permet de mettre à jour la liste des paquets installés.
`apt-get upgrade`	La commande « **`apt-get upgrade`** » permet de mettre à jour tous les paquets avec la dernière version disponible.

5.3.4 – Arrêt de la carte Raspberry

Commande	Description
`exit`	La commande « `exit` » permet de se déconnecter de la carte Raspberry.
`shutdown`	La commande « `shutdown` » permet de redémarrer ou d'éteindre la carte Raspberry : – `shutdown -r now` : permet de redémarrer la carte immédiatement. – `shutdown -h now` : permet d'éteindre la carte immédiatement.

6 – INTERFACE GRAPHIQUE

Après avoir configuré le Raspberry Pi une invite de commande apparaît à l'écran. Elle attend la saisie du nom d'utilisateur, en l'occurrence **pi** :

Saisie du nom d'utilisateur

```
raspberrypi login : pi
```

L'invite de commande demande ensuite un mot de passe. Saisir **raspberry** (la saisie n'est pas visible) :

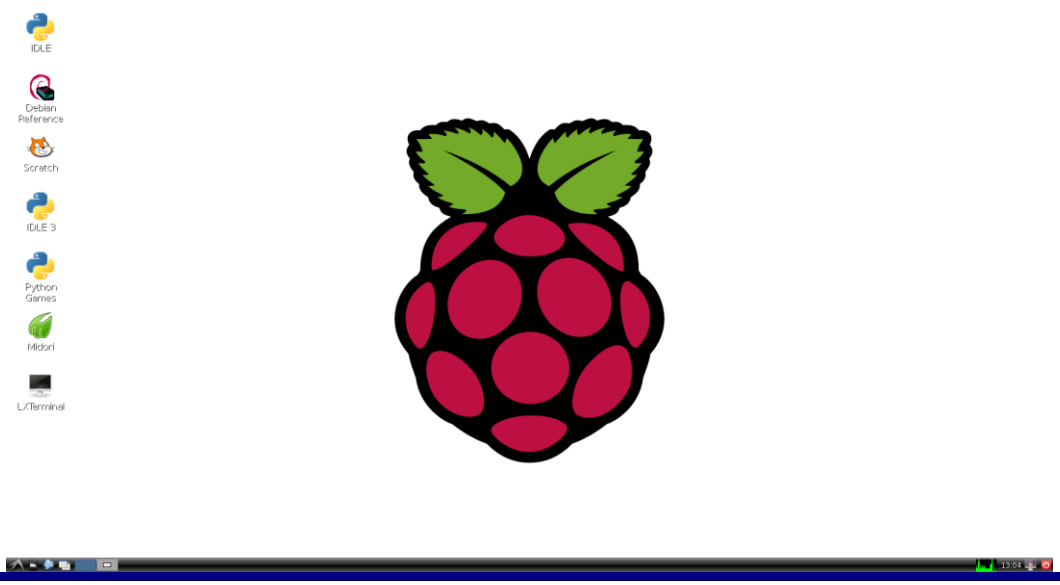
Saisie du mot de passe

```
Password: raspberry
```

Pour démarrer l'interface graphique il faut taper la commande « startx » et appuyer sur « Entrée » :

Interface graphique LXDE

```
pi@raspberrypi ~ $ startx
```



L'**interface graphique LXDE** (Lightweight X11 Desktop Environnement) fonctionne comme n'importe quel bureau Windows.

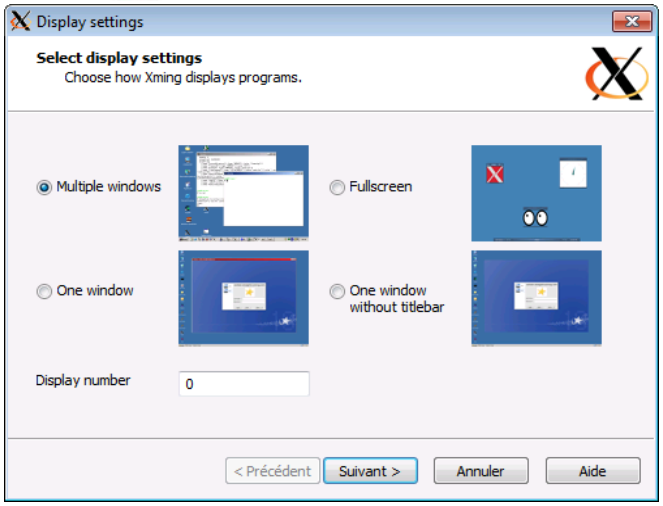
Les programmes sont accessibles à partir des éventuels raccourcis présents sur le bureau ou bien partir du menu auquel on accède grâce à l'icône « Menu » (comme pour le menu « Démarrer » de Windows). Les programmes sont classés en 7 catégories : « Accessoires », « Education », Internet », « Programmation », « Graphismes » et « Outils système ».

Le menu « **Programmation** » permet d'accéder aux interfaces de **développement des scripts « Python »** : **IDLE** (Python 2) et **IDLE3** (Python 3).

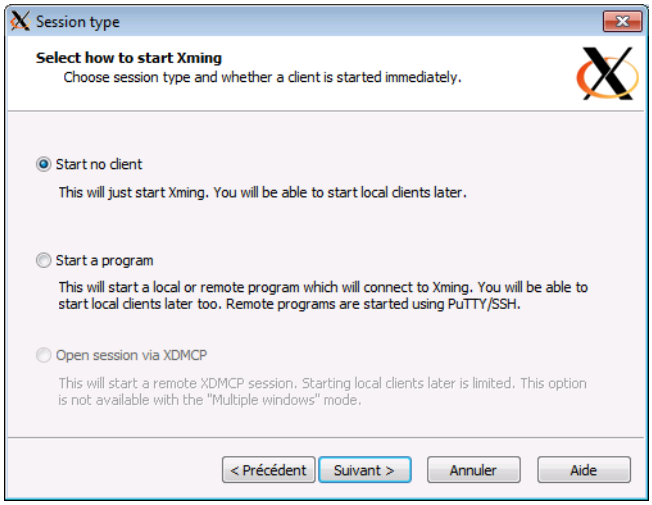
7 – CONTROLE A DISTANCE DU RASPBERRY PAR UNE LIAISON SSH

Il est possible de **contrôler à distance** le Raspberry Pi à l'aide d'un ordinateur distance au moyen d'une **liaison SSH**.

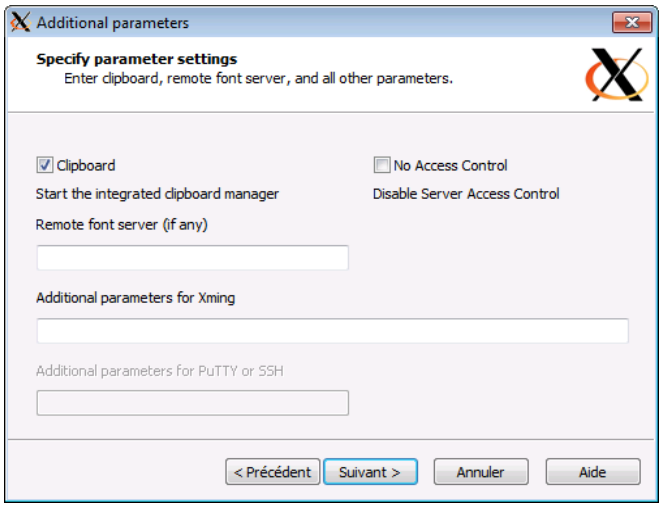
- Installer sur l'ordinateur le logiciel **Putty** : http://www.putty.org/
- Installer sur l'ordinateur le logiciel **Xming** : http://sourceforge.net/projects/xming/files/
- Lancer le logiciel **Xming Launch**. Choisir le **mode d'affichage** et cliquer sur « Suivant » :



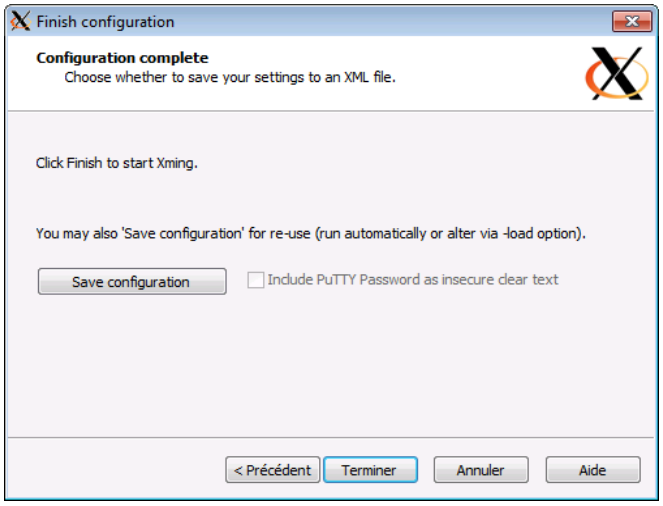
- Cliquer sur « Suivant » :



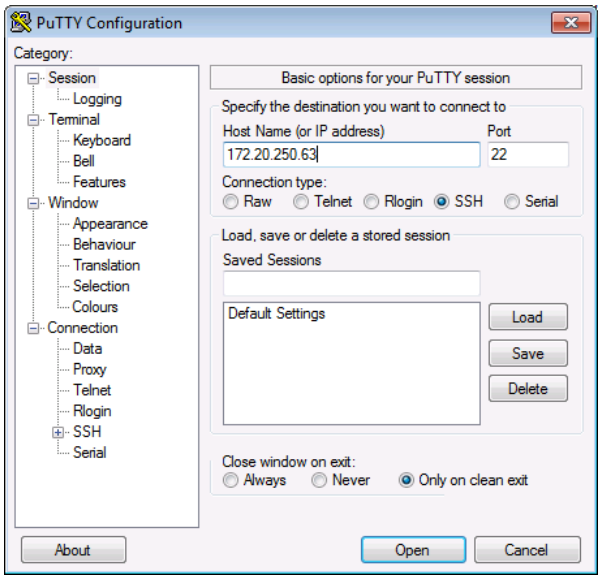
- Cliquer sur « Suivant » :



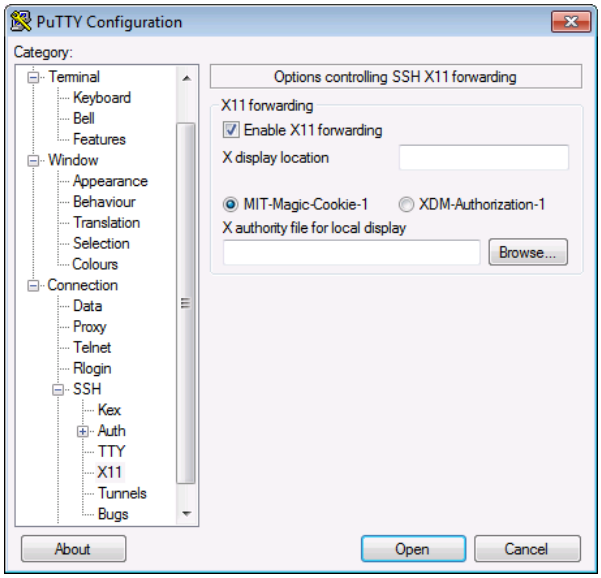
- Sauvegarder la configuration et cliquer sur « Terminer » :



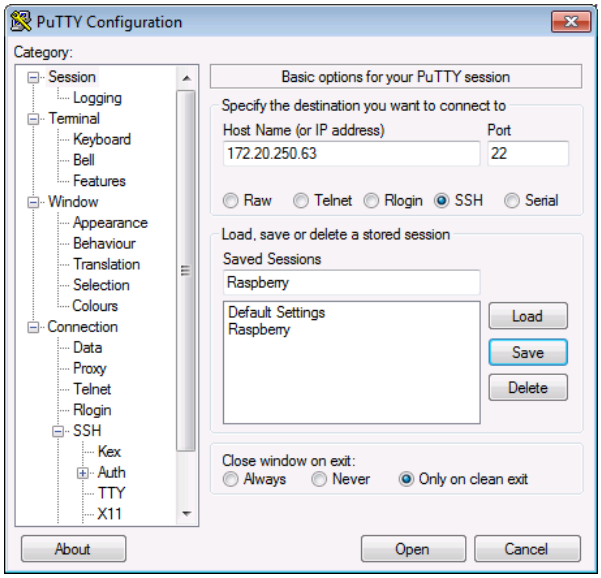
- Lancer le logiciel « **Putty** » . Entrer **l'adresse IP** de la carte Raspberry. Sélectionner, dans la fenêtre « Category », la rubrique « **SSH** » puis « **X11** ».



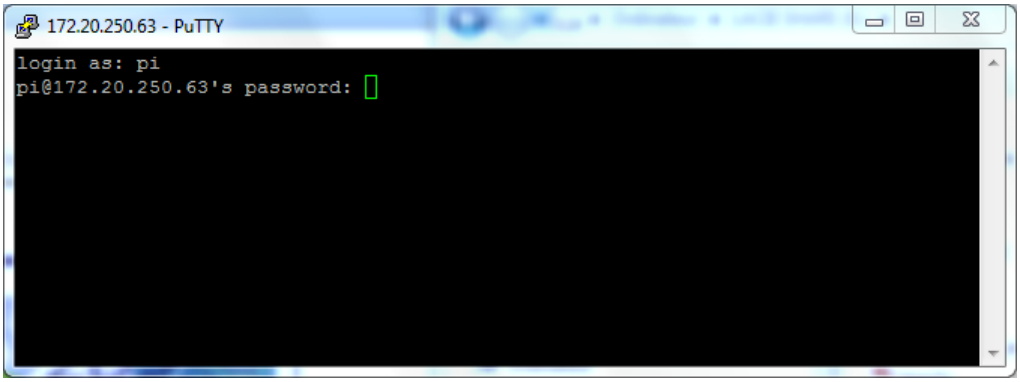
- Cocher la case « **Enable X11 forwarding** ». Sélectionner, dans la fenêtre « Category », la rubrique « **Session** ».



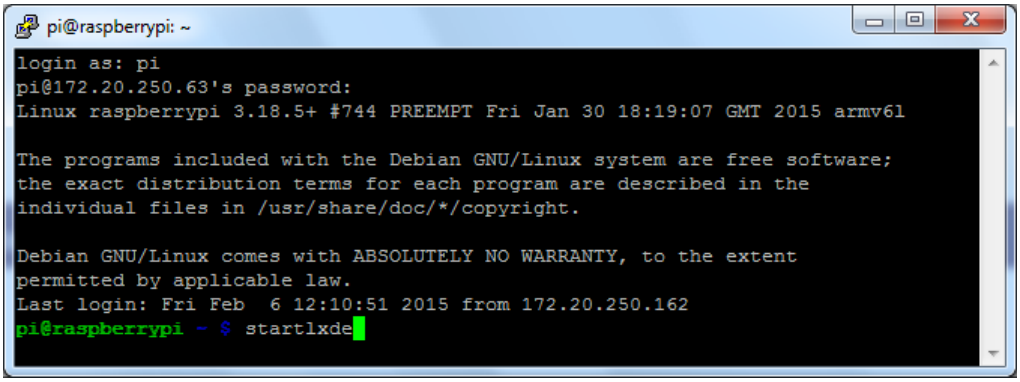
- Nommer la session dans la case « **Saved Session** ». Cliquer sur « **Save** » puis sur « **Open** ».



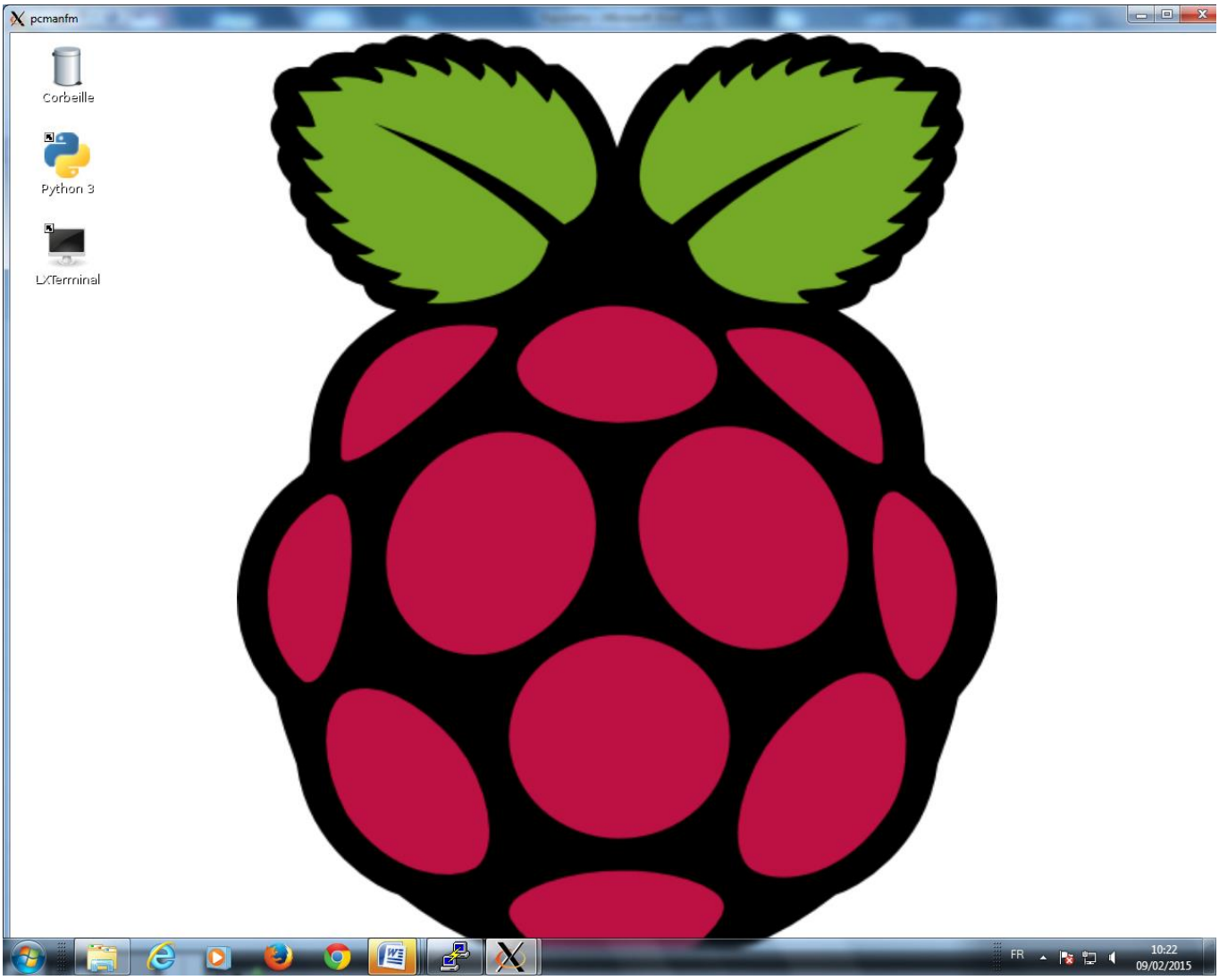
- Entrer le login « **pi** » et le nom d'utilisateur « **raspberry** » afin de connecter au Raspberry.



- Taper la commande « **startlxde** ».



- Il est maintenant possible de contrôler et d'accéder au Raspberry à partir d'un ordinateur distant.



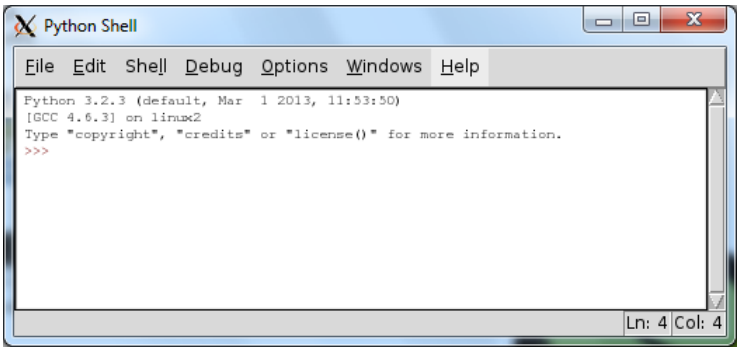
8 – LANGAGE PYTHON

8.1 – PRESENTATION DU LANGAGE PYTHON

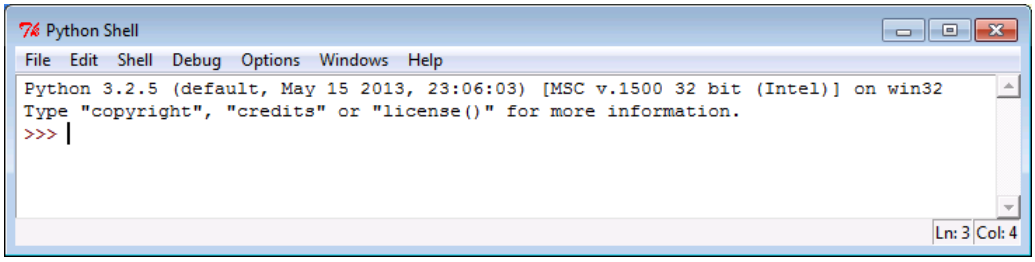
Le langage Python est un **langage informatique de haut niveau interprété**, c'est-à-dire que le code est exécuté ligne à ligne (voir le document ressource « Le langage Python »).

L'interface de développement d'un script Python est l'application **IDLE** pour la version 2 de python ou **IDLE3** pour la version 3.

Fenêtre d'accueil de IDLE (ou Shell Python) dans l'interface graphique LXDE

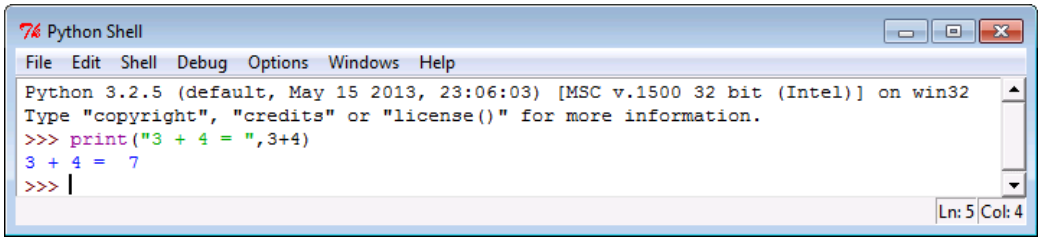


Fenêtre d'accueil de IDLE (ou Shell Python) sous Windows



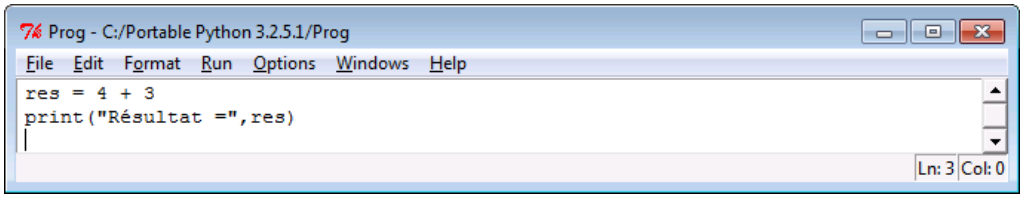
Lorsque IDLE est lancé, la fenêtre Shell est mode **interpréteur interactif** (reconnaissable aux trois chevrons >>>), c'est-à-dire que des lignes de codes python peuvent être exécutées immédiatement.

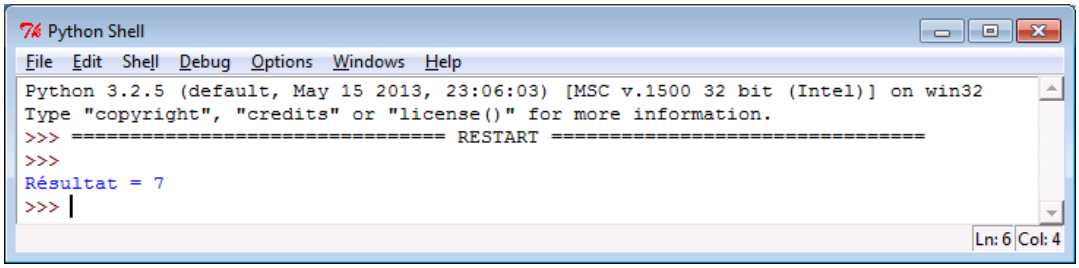
Interpréteur interactif



Pour **éditer un programme** (ou un script) Python, il faut ouvrir une **fenêtre d'édition**. Dans la fenêtre Shell, il faut sélectionner le menu « **File** » puis cliquer sur « **New Window** ». Le programme peut être édité dans la fenêtre qui s'est ouverte. Un fois le programme édité et enregistré (au format **.py**), il peut être exécuté par l'intermédiaire du menu « **Run** » en cliquant sur « **Run Module** » ou en appuyant sur la **touche « F5 »**. Le résultat de l'exécution du programme s'affiche dans la fenêtre.

Edition et exécution d'un script Python





Les programmes Python sont enregistrés dans la carte SD du Raspberry PI afin d'être exécuter ultérieurement. Il est conseillé de créer un répertoire « prog_python » dans l'environnement « /home/pi ». Les différents programmes seront enregistrés dans ce répertoire.

8.2 – CREATION D'UN FICHIER EXECUTABLE

Pour créer un programme python exécutable par la seule saisie de son nom il faut :

- Indiquer au programme **où se trouve l'interpréteur Python** dans la machine. Pour cela il faut ajouter la ligne de code « `# !usr/bin/env python` » au début du programme Python.
- Ouvrir une fenêtre LXTerminal et saisir « `pi@raspberrypi ~ $ cd prog_python` » pour passer dans le répertoire « programmes_python ».
- Saisir « `pi@raspberrypi ~ /prog_python $ chmod +x fichier.py` » pour rendre le fichier « fichier.py » exécutable.
- Copier le fichier « nom_fichier.py » dans le répertoire « /usr/local/bin » pour l'exécuter à partir de n'importe quel répertoire dans avoir à saisir le chemin d'accès au fichier. Pour cela saisir « `pi@raspberrypi ~ /prog_python $ sudo cp fichier.py /usr/local/bin` ».
- Pour ne pas avoir à saisir l'extension .py lors de l'exécution du programme Pyhton, saisir « `pi@raspberrypi ~ /prog_python $ sudo mv /usr/local/bin/fichier.py /usr/local/bin/fichier` ».

La saisie du nom du fichier depuis n'importe quel répertoire machine conduira à l'exécution du programme Python.

8.3 – BIBLIOTHEQUE RPI.GPIO

La bibliothèque **RPi.GPIO** met à disposition les fonctions permettant de **gérer les entrées/sorties du port GPIO**.

Cette bibliothèque est **installée par défaut** sur la distribution Raspbian. Il cependant nécessaire de vérifier que la dernière version de la librairie est installée.

Pour **installer cette bibliothèque** saisir dans une fenêtre LXTerminal, les commandes :

```
pi@raspberrypi ~ /prog_python $ apt-get update
pi@raspberrypi ~ /prog_python $ apt-get install python-rpi.gpio
```

Pour **vérifier que la bibliothèque** est correctement installée saisir les commandes ci-dessous. Si aucun message d'erreur ne s'affiche, la bibliothèque est bien installée.

```
pi@raspberrypi ~ /prog_python $ sudo python
>>> import RPi.GPIO as GPIO
```

8.3.1 – Fonction GPIO.setmode()

La fonction `GPIO.setmode()` permet de **repérer les entrées/sorties du port GPIO** par le numéro de la broche du connecteur GPIO « `GPIO.setmode(GPIO.BOARD)` » ou bien par le numéro de la broche du processeur Broadcom « `GPIO.setmode(GPIO.BCM)` ».

8.3.2 – Fonction GPIO.setup() :

La fonction `GPIO.setup()` permet de configurer le **sens de fonctionnement** entrée (`GPIO.IN`) ou sortie (`GPIO.OUT`) du numéro de port spécifié en paramètre.

Exemples

```
GPIO.setup(23, GPIO.OUT)    # Le port 23 est configuré en sortie

GPIO.setup(25, GPIO.IN)     # Le port 25 est configuré en entrée
```

Il est possible de définir un **état initial** pour un port configuré en sortie.

Exemples

```
# L'état initial du port 23 configuré en sortie est le niveau logique 0
GPIO.setup(23, GPIO.OUT, initial=GPIO.LOW)

# L'état initial du port 23 configuré en sortie est le niveau logique 1
GPIO.setup(25, GPIO.IN, initial=GPIO.HIGH)
```

Les lignes de port du processeur Broadcom sont équipées de résistances de tirage qu'il est possible de désactiver (`GPIO.PUD_OFF`) de configurer en résistances de pull-up (`GPIO.PUD_UP`) ou en résistances de pull-down (`GPIO.PUD_DOWN`).

Exemples

```
# Résistance de tirage désactivée sur le port 25 configuré en entrée
GPIO.setup(25, GPIO.IN, pull_up_down=GPIO.PUD_OFF)

# Résistance de tirage = Pull_up sur le port 25 configuré en entrée
GPIO.setup(25, GPIO.IN, pull_up_down=GPIO.PUD_UPF)

# Résistance de tirage = Pull_down sur le port 25 configuré en entrée
GPIO.setup(25, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

8.3.3 – Fonction GPIO.output()

La fonction `GPIO.output()` permet de **placer la ligne de port** spécifiée (configurée en sortie) à **l'état haut** (`GPIO.HIGH` ou `True`) ou à **l'état bas** (`GPIO.LOW` ou `False`).

Exemples

```
GPIO.output(23, GPIO.LOW)  # Le port 23 est placé au NL0

GPIO.output(23, False)     # Le port 23 est placé au NL0

GPIO.output(23, GPIO.HIGH) # Le port 23 est placé au NL1

GPIO.output(23, True)      # Le port 23 est placé au NL1
```

8.3.4 – Fonction GPIO.input()

La fonction `GPIO.input()` permet de **lire l'état de la ligne de port** spécifiée (configurée en entrée).

Exemple

```
Entree = GPIO.input(25)    # Lire l'état du port 23 est l'affecter à Entree
```

8.3.5 – Fonction GPIO.add_event_detect()

La fonction `GPIO.add_event_detect()` permet de valider la **détection** automatique d'un **front montant** (`GPIO.RISING`), d'un **front descendant** (`GPIO.RISING`) ou **les deux** (`GPIO.BOTH`) sur la **ligne de port** spécifiée (configurée en entrée).

Exemples

```
GPIO.add_event_dectect(25, GPIO.RISING)     # Détection front montant
GPIO.add_event_dectect(25, GPIO.FALLING)    # Détection front descendant
GPIO.add_event_dectect(25, GPIO.BOTH) # Détection front montant ou descendant
```

8.3.6 – Fonction GPIO.remove_event_detect()

La fonction `GPIO.remove_event_detect()` permet **d'arrêter la détection automatique d'un front** qui aurait été préalablement validée sur la ligne de port spécifiée (configurée en entrée).

Exemple

```
GPIO.remove_event_dectect(25)    # Arrêt de la détection d'un front
```

8.3.7 – Fonction GPIO.event_detect()

La fonction `GPIO.event_detect()` permet une **prise de décision lors de la détection d'un front** validée sur la ligne de port spécifiée. Il faut que la détection automatique d'un front soit préalablement validée.

Exemple

```
if GPIO.event_dectect(25) :      # Si un front est détecté sur le port 25
     Bloc d'instructions à exécuter
Suite du programme
```

8.3.8 – Fonction GPIO.cleanup()

La fonction `GPIO.cleanup()` permet de **réinitialiser toutes les lignes** d'entrées/sorties du port GPIO à leur **état par défaut** : Entrée sans résistance de tirage et sans détection automatique de front.

Exemple

```
GPIO.cleanup()
```