

Vous voulez développer vos propres programmes sur TI Nspire (CX) (CAS) ?

Vous en avez marre du TI-Basic qui n'a pas d'instructions graphiques ?

Vous êtes sur Windows 10 et vous ne savez pas où vous n'arrivez pas à installer le SDK (Kit de développement logiciel) pour Ndless ?

Alors ce tutoriel est fait pour vous !

1 : Avant toutes choses il va falloir regarder si la version de votre Windows 10 est compatible pour installer le Sous-système Linux. Pour le vérifier allez dans les Paramètres -> Système -> Information Système.

About



HP Pavilion Notebook

PC name LAPTOP-8NUBRTR0

Rename this PC

Organisation WORKGROUP

[Connect to work or school](#)

Edition Windows 10 Home

Version 1703

OS Build 15063.674

Product ID

Processor AMD A9-9410 RADEON R5, 5 COMPUTE CORES  
2C+3G 2.90 GHz

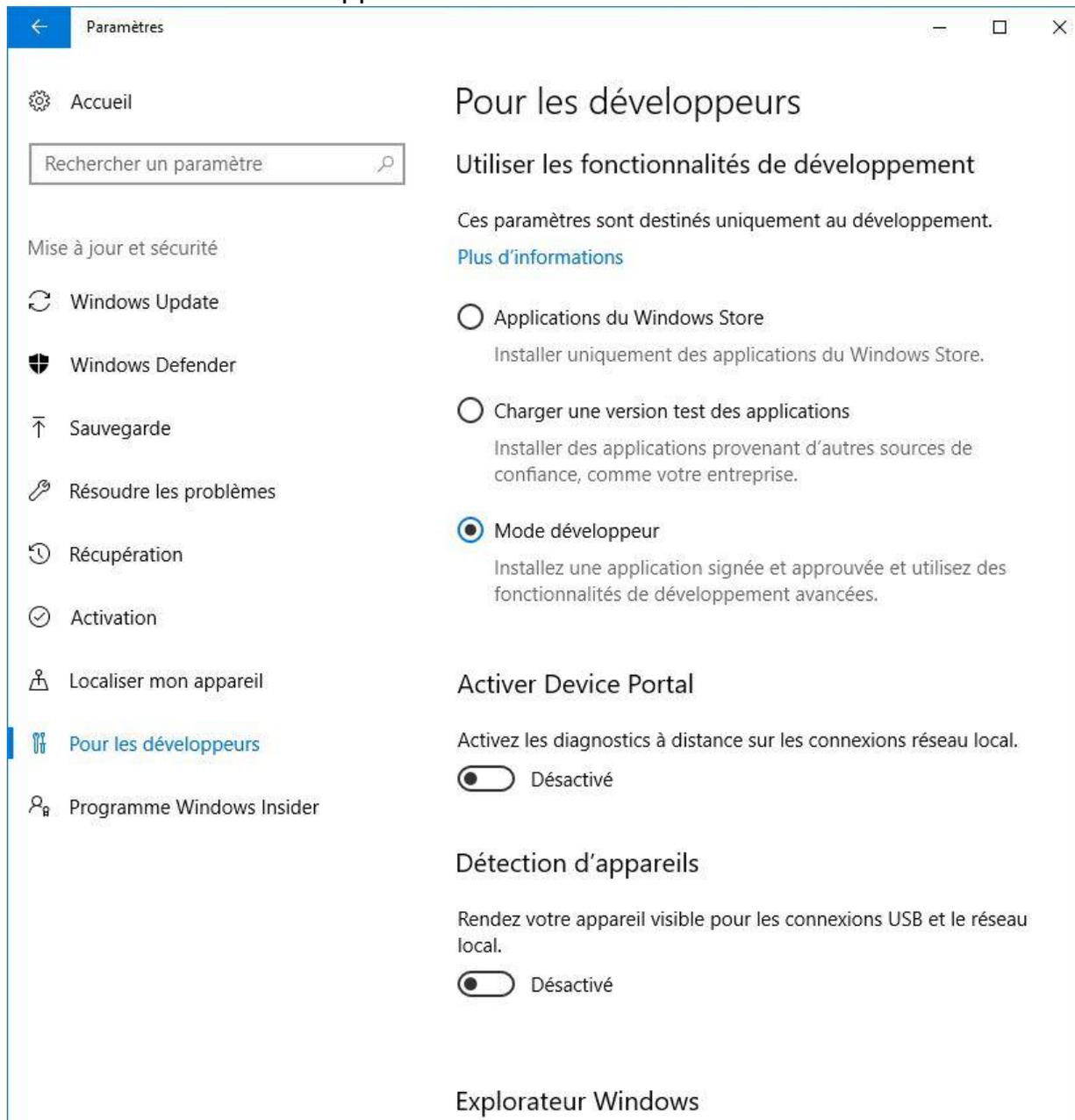
Installed RAM 8.00 GB (7.45 GB usable)

System type 64-bit operating system, x64-based processor

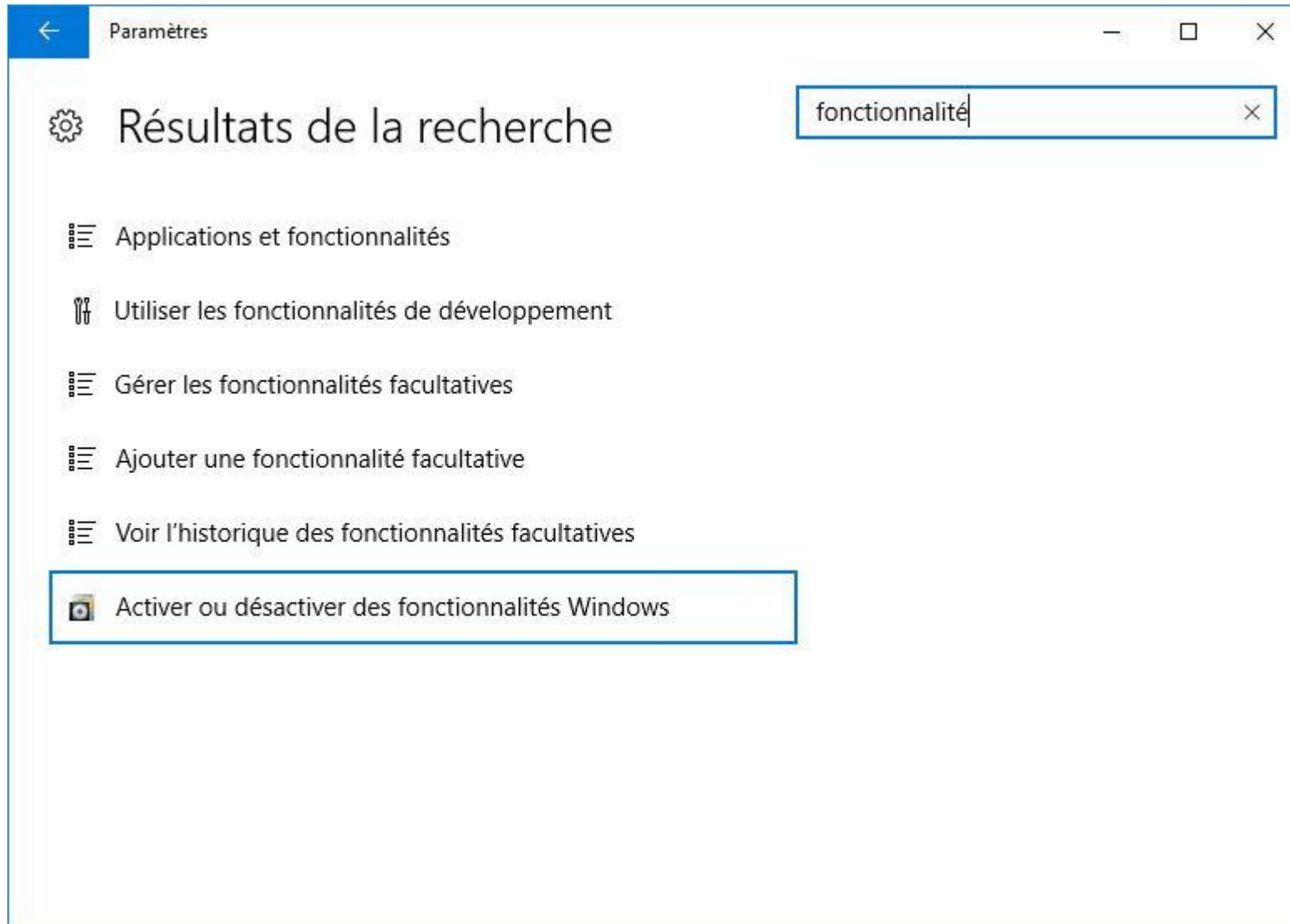
Pen and touch No pen or touch input is available for this display

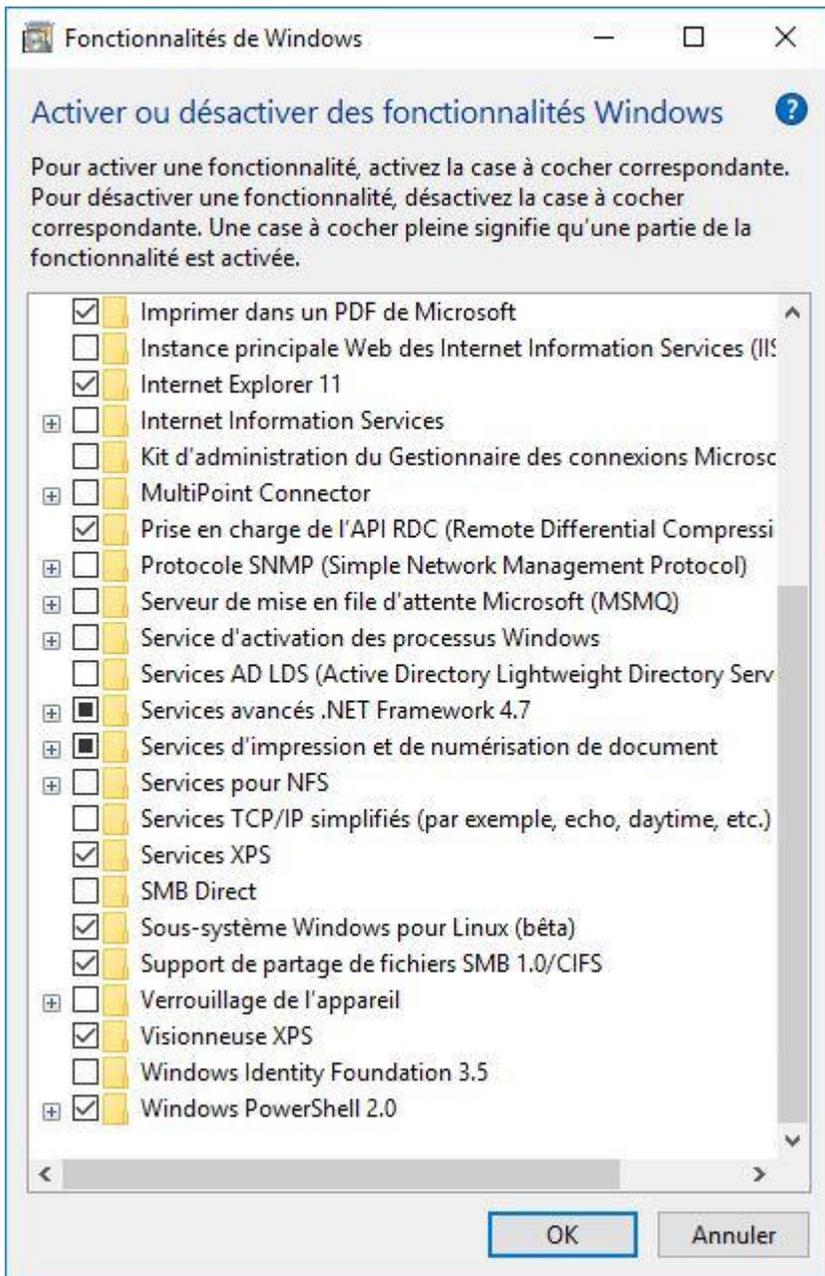
Vous devez avoir une version de Build supérieure ou égale à 14393 et Windows 10 doit être la version 64bits

2 : Ensuite rendez-vous dans les paramètres -> Mise à jour et sécurité et dans le menu « Pour les développeurs » cochez le bouton « Mode développeur »

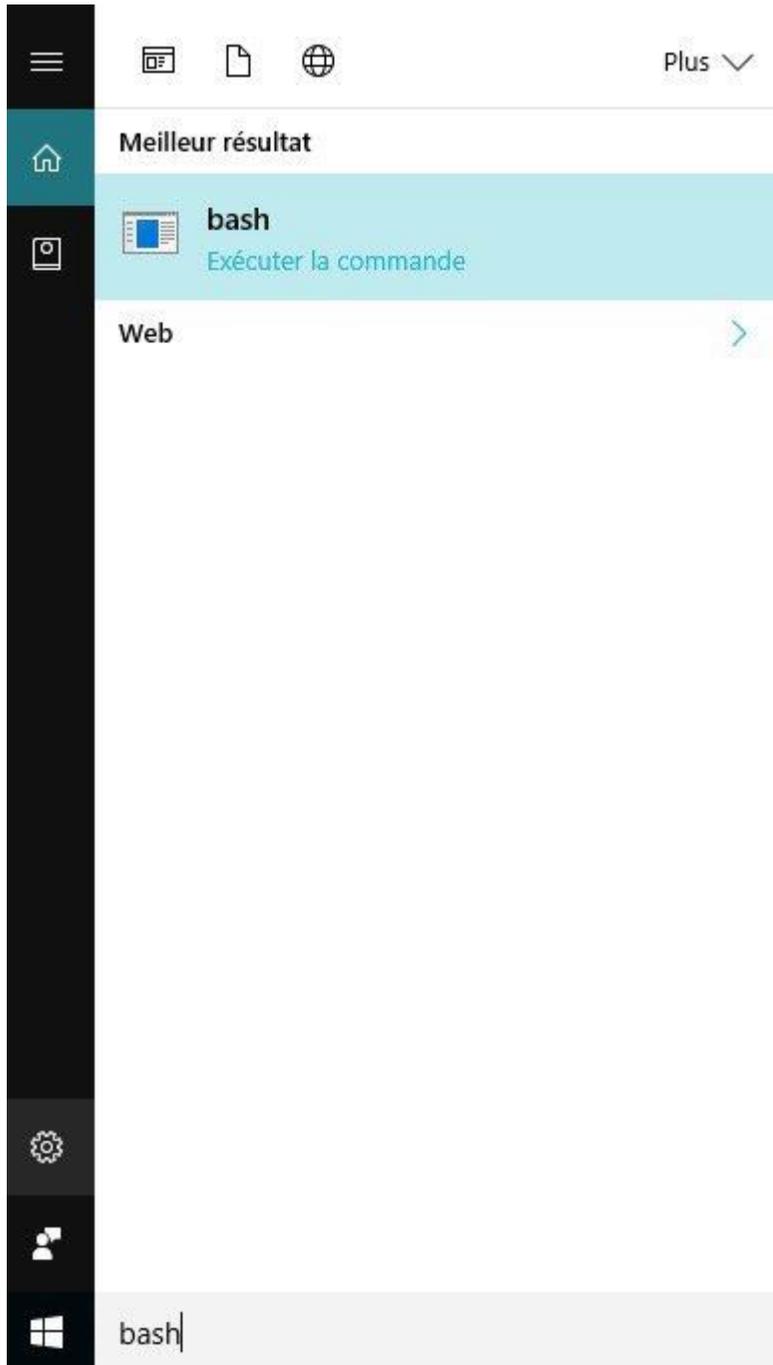


3 : Ensuite on va activer le « Sous système Linux » de Windows. Pour cela tapez « Fonctionnalités » dans la barre de recherche et cliquez sur « Activer ou désactiver des fonctionnalités Windows »



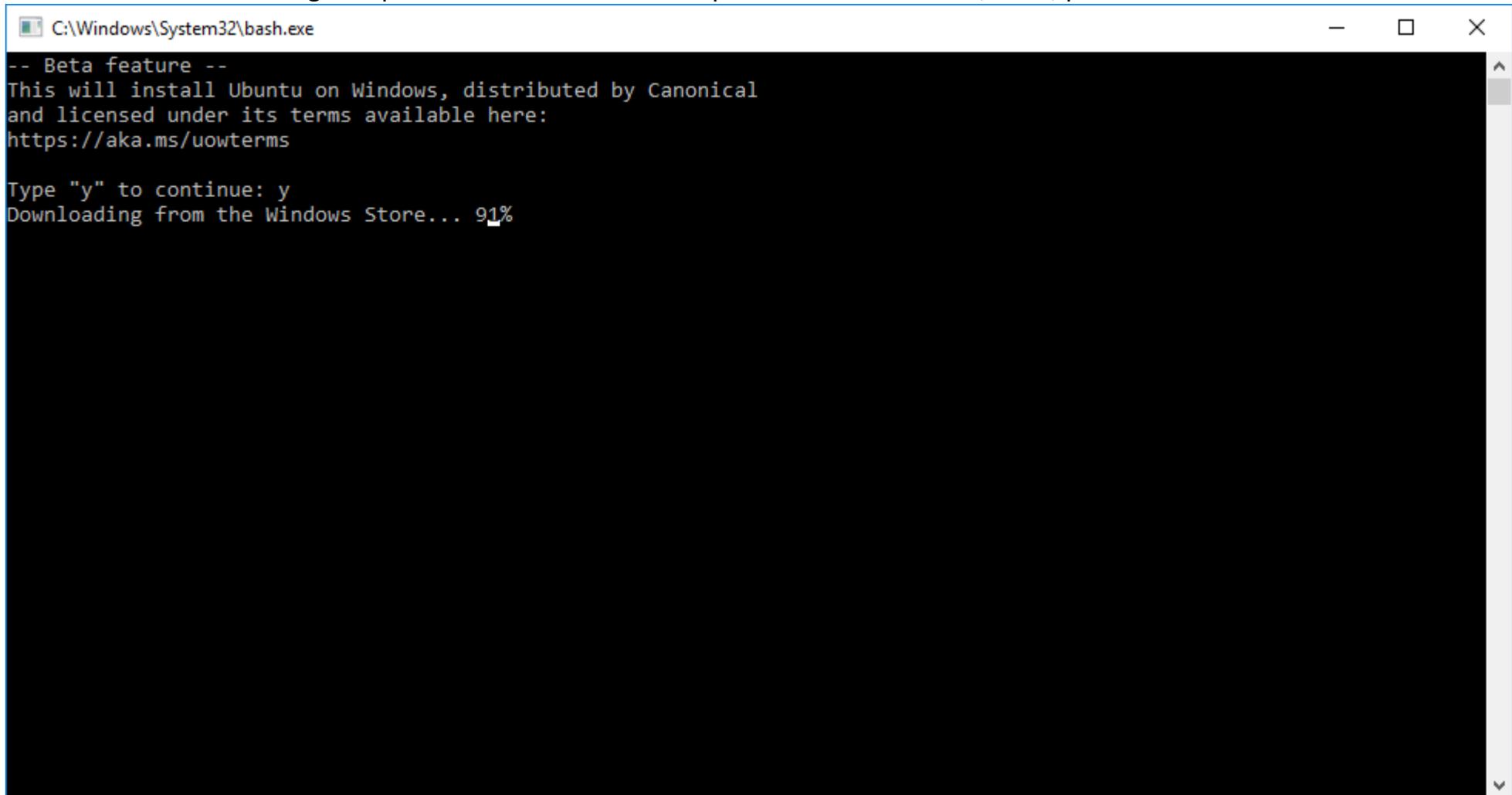


Vous verrez alors apparaître cette fenêtre.  
Cochez la case « Sous-système Windows pour Linux » et  
Appuyez « OK ».  
Votre ordinateur devra ensuite redémarrer.



Maintenant dans la barre de recherche, tapez « bash » et lancez-le.

Une fenêtre va s'ouvrir vous demandant d'accepter la licence Canonical, donc appuyez sur la touche « o » pour continuer et vous verrez Ubuntu se télécharger depuis le Windows Store. Cela peut durer un moment, donc, patientez.

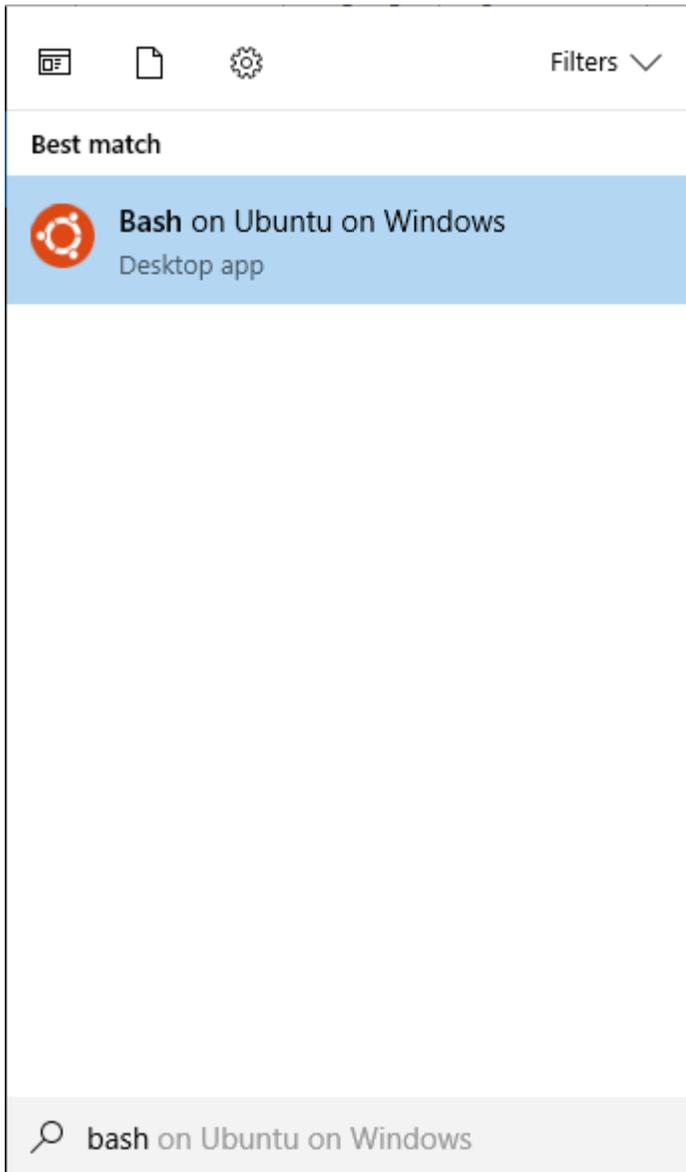
A screenshot of a Windows command prompt window titled "C:\Windows\System32\bash.exe". The window has a black background with white text. The text displayed is: "-- Beta feature --", "This will install Ubuntu on Windows, distributed by Canonical", "and licensed under its terms available here:", "https://aka.ms/uowterms", "Type 'y' to continue: y", and "Downloading from the Windows Store... 91%". The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

```
C:\Windows\System32\bash.exe
-- Beta feature --
This will install Ubuntu on Windows, distributed by Canonical
and licensed under its terms available here:
https://aka.ms/uowterms

Type "y" to continue: y
Downloading from the Windows Store... 91%
```

Dans le cas où bash vous fait une erreur à l'ouverture, par exemple parce que vous essayez de le réinstaller ou tout autre raison, il existe une méthode simple et rapide pour corriger le problème.

Clic droit sur le bouton démarrage de la barre des tâches, démarrez « Windows PowerShell(Admin) » et entrez les commandes suivantes. **lxrun /uninstall /full** et tapez « Entrée », à la question posée tapez « y » ou « o » cela dépend de la langue du système. Ensuite entrez la commande **lxrun /install /y** et tapez « Entrée », le sous-système Ubuntu va se télécharger et s'installer.



Une fois que c'est installé, vous pouvez lancer Bash depuis le menu de Windows.

Ce Bash permet de faire tout ce qui est possible dans un bash sous Ubuntu.

Un exemple, si vous voulez faire défiler du code en mode Matrix, tapez :  
`sudo apt-get install cmatrix`

Patientez pendant l'installation, et une fois terminé tapez : `cmatrix`

Passons maintenant à l'installation du SDK.

Pour que le SDK fonctionne correctement, vous devez installer toutes les dépendances nécessaires : git, gcc avec le support c++ etc. ...  
Pour cela tapez cette longue ligne dans le bash :

```
sudo apt-get install git build-essential binutils libgmp-dev libmpfr-dev libmpc-dev zlib1g zlib1g-dev zlib1g-dbg libboost-dev libboost-program-options-dev libboost-program-options1.58-dev libboost-program-options1.58.0 libboost1.58-dev wget texinfo libpython2.7-dev bison flex php7.0-cli
```

Nous allons maintenant récupérer la dernière version du code du SDK Ndless.  
Mais avant de le télécharger placez-vous dans le bon répertoire avec la commande :

```
cd /mnt/c/Users/NomUtilisateur
```

Ensuite tapez :

```
git clone --recursive https://github.com/ndless-nspire/Ndless.git
```

Cela va automatiquement télécharger le SDK, et le mettre dans le répertoire principal dans un dossier Ndless.

Maintenant, nous avons besoin de modifier le fichier build\_toolchain.sh.

Sur votre ordinateur et non dans le bash, rendez-vous dans le dossier dans lequel vous avez cloné le SDK. Si vous n'avez pas modifié ce dossier avant d'exécuter la commande git, il devrait se trouver dans le dossier C:\Users\... sous la forme d'un dossier Ndless (les ... sont à remplacer par votre nom d'utilisateur).

Allez dans le dossier Ndless, puis ndless-sdk et enfin toolchain. Là, vous devriez voir un fichier build\_toolchain.sh. Ouvrez ce fichier avec un éditeur de texte.

Vous allez devoir modifier la ligne 31 : **OPTIONS\_GDB="--target=\${TARGET} --prefix=\${PREFIX} --enable-interwork --enable-multilib --disable-werror --with-python"**

Enlevez le **--with-python**

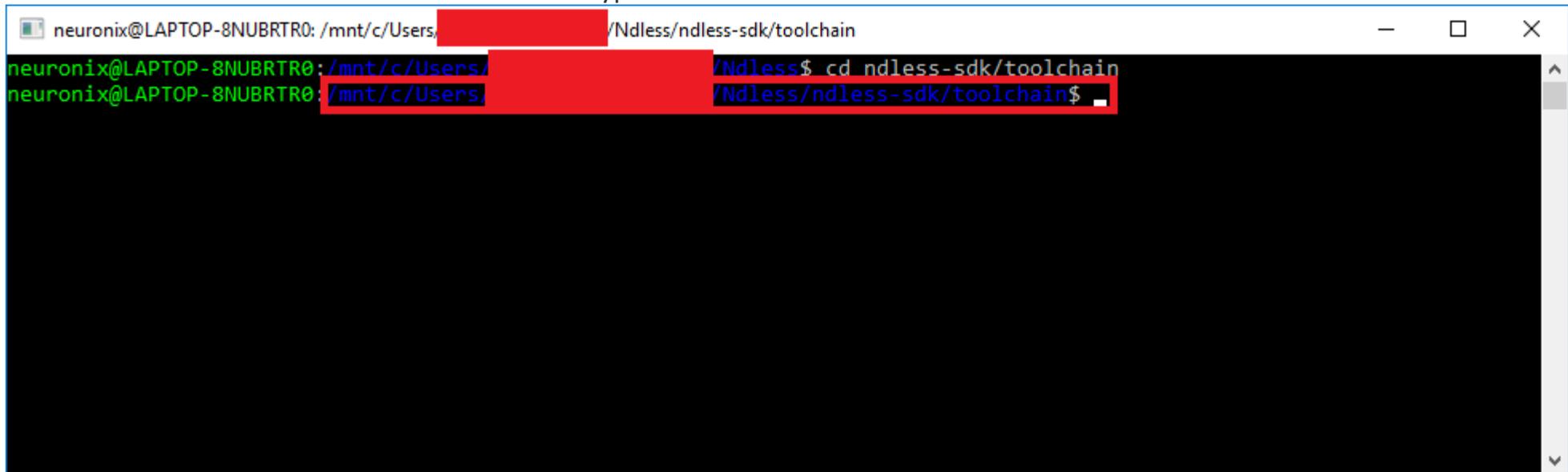
Maintenant, allez à la ligne 51 : `if ! gcc -lpython2.7 test.c -o test; then error=1; echo 'libpython2.7 (python-devel/python2.7-dev) dependency seems to be missing!'; fi`

Rajoutez un '#' devant (sans les ' bien sûr).

Cela devrait transformer la ligne en un commentaire, donc non exécutée.

Je précise que dans le sous-système Linux de Windows10, votre disque principal C'est dans le dossier `/mnt/c` dans le bash. Donc si vous voulez accéder au dossier Users de votre disque C, tapez `cd /mnt/c/Users` et non `cd C/Users` ou tout autre chemin.

Vous devriez maintenant avoir une fenetre de e type.



```
neuronix@LAPTOP-8NUBRTR0: /mnt/c/Users/[redacted]/Ndless/ndless-sdk/toolchain
neuronix@LAPTOP-8NUBRTR0: /mnt/c/Users/[redacted]/Ndless$ cd ndless-sdk/toolchain
neuronix@LAPTOP-8NUBRTR0: /mnt/c/Users/[redacted]/Ndless/ndless-sdk/toolchain$
```

Si ce n'est pas le cas, rendez-vous dans ce dossier (les carrés rouges sont le nom d'utilisateur).

Maintenant, retournez dans votre bash, et allez dans le sous-dossier toolchain : `cd Ndless/ndless-sdk/toolchain`

Exécutez maintenant la commande `./build_toolchain.sh`.

L'opération risque d'être longue, donc prenez votre mal en patience .

Une fois terminé, vous devriez avoir des lignes de ce type affichées:

```
make[11]: Nothing to be done for 'install-data-am'.
make[11]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib/import'
make[10]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib/import'
make[9]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib/import'
make[8]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib/import'
make[7]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib'
make[6]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib'
make[5]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb/build-gnulib'
make[4]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb'
make[3]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb'
make[2]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build/gdb'
make[1]: Nothing to be done for 'install-target'.
make[1]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/build'
Done!
Don't forget to add '/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/install/bin' to your $PATH along with
/mnt/c/Users/neuronix/Ndless/ndless-sdk/toolchain/./bin.
```

Bien sûr, les `/mnt/c/Users/neuronix/Ndless/` sont à remplacer par le chemin vers votre dossier Ndless.  
Si ce n'est pas le cas, vérifiez que vous avez tout bien fait comme il faut.

Fermez maintenant la console bash.

Recherchez variables d'environnement à l'aide de l'outil Recherché de Windows 10, et ouvrez Modifier les Variables d'Environnement Système. Une fenêtre s'ouvre, cliquez sur Variables d'Environnement. Dans le cadre en haut, cliquez sur Path, et enfin, cliquez sur Edit. Maintenant ajoutez deux nouveaux Chemins:

[Chemin\_vers\_ndless]\ndless-sdk\toolchain\install\bin\ et [chemin\_vers\_ndless]\ndless-sdk\bin\.

Dans mon cas se sera **C:\Users\neuronix\Ndless\ndless-sdk\toolchain\install\bin\** et **C:\Users\neuronix\Ndless\ndless-sdk\bin\**

N'oubliez pas que dans ce cas, ce sont des chemins Windows, donc '\' et non '/' (antislash et non slash).

Fermez toute les fenêtres en appuyant sur Ok.

Vous pouvez désormais ré-ouvrir le bash.

Allez dans le dossier Ndless, donc dans mon cas, juste à exécuter `cd Ndless`

Et enfin, tapez `make`.

Si vous avez tout suivi jusqu'à maintenant, il ne devrait pas y avoir de problème, et aucune erreur.

Si tout c'est bien passé, vous devriez avoir ces 3 dernières lignes affichées, avec le chemin qui change bien sûr:

```
make[3]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless/src/installer-4.5'
```

```
make[2]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless/src'
```

```
make[1]: Leaving directory '/mnt/c/Users/neuronix/Ndless/ndless'
```

Pour vérifier que tout est fonctionnel, tapez **nspire-gcc**.

Vous devriez voir ces lignes apparaitre:

```
arm-none-eabi-gcc: fatal error: no input files
```

```
compilation terminated.
```

Si vous avez ça, c'est bon, tout est prêt, vous pouvez désormais créer et compiler vos propres projets!