

Programmer en C sur votre TI Nspire :

Tout d'abord, il faut disposer de certaines connaissances et objets pour pouvoir programmer votre TI Nspire. Vous devez :

- Avoir lu les deux premières parties au moins du tutoriel [Apprenez à programmer en C](#) de M@theo21.
- Avoir une calculatrice TI Nspire (de type CAS ou pas).
- Avoir un ordinateur ainsi que le câble de connexion entre celui-ci et la calculatrice.
- Avoir installé le logiciel de transfert TI-PC sur votre ordinateur.
- Avoir installé Code Blocks de préférence ou simplement MinGW sur votre ordinateur.

Sachez aussi que si vous avez l'OS (Système d'exploitation) 1.1 installé sur votre TI Nspire cela vous facilitera beaucoup la vie mais ce n'est pas obligatoire. Pour le savoir, allez sur la page [A propos du menu Infos système](#).

Bien ! Nous allons maintenant pouvoir entrer dans le coeur du tutoriel !

Débrider sa calculatrice

Pour l'instant, votre calculatrice TI Nspire est totalement bridée. Elle ne peut lire que des classeurs banals composés de calculs , de graphiques, ou d'autres applications purement mathématiques. Nous allons changer cela grâce à un merveilleux exécutable nommé Ndless, qui vous permettra d'ouvrir votre calculatrice au C.

Comme je suis gentil, je vais vous donner le lien de téléchargement :

[Lien de téléchargement de Ndless 1.1](#)

Décompressez le fichier avec un logiciel comme WinRar ou 7-Zip.

Ici vous allez IMPERATIVEMENT avoir besoin de l'OS 1.1 sur votre calculatrice. Si vous ne l'avez pas, vous allez devoir le chercher sur Internet.

Je n'ai malheureusement pas le droit de vous dire où le trouver suite à une lettre qu'à envoyée TI au site Ti-Bank.fr qui demandait à ce dernier d'enlever le lien de téléchargement de cet OS pour Copyright. Cependant j'ai entendu dire que la justice avait déclaré que les réclamations de TI n'étaient pas justifiées.

De toute façon en cherchant bien vous le trouverez cet OS.

Après que vous ayez acquis l'OS 1.1 (ou non si vous l'avez déjà), nous allons pouvoir installer Ndless sur votre calculatrice.

Entrez dans le dossier ndless que vous avez décompressé auparavant.

Si vous n'avez pas l'OS et que vous l'avez trouvé, créez un dossier nommé exactement userfiles et copiez l'OS à l'intérieur en le renommant exactement tinspire_1.1.tno (.tno étant l'extension bien sûr).

Ensuite allumez votre calculatrice, branchez le câble et double-cliquez sur l'exécutable *install-ndless*. Suivez

les instructions qui s'affichent. Attention, c'est en Anglais mais vous n'aurez je pense aucunes difficultés à comprendre.

Attention lorsque la calculatrice redémarre et vous demande la langue à utiliser, choisissez Anglais sinon Ndless ne s'installera pas. Vous pourrez changer cette option plus tard en allant sur la page Réglages du système du menu Infos système.

Voilà ! Si tout s'est bien passé, votre calculatrice devrait être débridée. Pour le savoir allez dans le menu Mes classeurs, dossier ndless, et cliquez sur demo. Si la calculatrice arrive à ouvrir ce classeur : c'est gagné !

Sur le PC

Bien ! Maintenant il va nous falloir deux programmes pour configurer un environnement de développement en C. Ces deux programmes se nomment MSYS et Yagarto. Ils vont vous permettre de compiler votre code C.

MSYS

Allons-y ! Tout d'abord, téléchargeons le programme. Voici un lien direct :

[Lien de téléchargement de MSYS](#)

Lancez le programme, c'est un simple executable de type suivant-suivant-quitter...

Cependant à un moment une console s'ouvre. Répondez 'yes' à la première question en tapant y. De même pour la deuxième question.

Pour la troisième, ça se corse . Il faut dire au programme où votre version de MinGW est installée. Si vous avez bien Code Blocks vous devriez l'avoir dans le dossier d'installation de ce dernier.

Attention ! Il faut écrire le chemin sous la forme C:/xxxx/xxxx/xxxx et non C:\xxxx\xxxx\xxxx.

Deuxième problème ! La console n'accepte pas d'espaces dans le nom des dossiers !

Aïe ! Pas d'espace ! Si vous avez installé Code Blocks dans le répertoire par défaut, alors il est dans *Program Files*. Et qui dit *Program Files* dit espace !

Vous allez donc devoir déplacer le dossier MinGW autre part ! L'idéal serait de le mettre à la racine, donc dans C:.

Au final le chemin sera donc, en tenant compte des deux problèmes : C:/MinGW. Ouf ! Appuyez sur entrée jusqu'à ce que la console se ferme. Puis fermez le fenêtre d'installation. Bon, programme suivant.

Yagarto

Passons au programme Yagarto ! Ce sera beaucoup plus simple ici .

Voici le lien pour le télécharger :

[Lien de téléchargement de Yagarto](#)

Une fois téléchargé (le téléchargement prends un peu plus de temps que MSYS), lancez le programme.

Je vous laisse faire, c'est un programme d'installation tout simple (sans console cette fois-ci). Cependant veillez à avoir bien coché la case Add Yagarto to the PATH variable.

Voilà c'est presque fini !

Ndless

Il reste une dernière étape à passer : ajouter les programmes de Ndless à la variable d'environnement.

Pour ce faire, suivez ces instructions :

- Cliquez sur Menu Démarrer,

- Cliquez droit sur Poste de travail/Ordinateur,
- Cliquez sur Propriétés,
- Cliquez sur Paramètres système avancés,
- Cliquez sur le bouton Variable d'environnement,
- Dans la liste Variables système, double-cliquez sur la variable Path,
- Enfin, ajoutez un point-virgule dans la deuxième zone de texte et écrivez le chemin de l'emplacement du dossier *bin* de Ndless. Pour moi c'est C:\Users\rob36\Downloads\ndless\ndless-v1.1\bin. Validez.

Enfin nous avons fini ! Nous allons pouvoir commencer à programmer !

Compiler un programme

Super ! Maintenant que nous avons les outils nécessaires, nous allons pouvoir programmer mais tout d'abord, il faut tester ces outils pour voir s'ils sont bien installés.

Pour ce faire, eh bien nous allons compiler la demo de Ndless. Vous vous rappelez ? Le classeur qui servait à vérifier si Ndless était bien installé.

Donc pour compiler, il va falloir lancer le programme MSYS que vous avez installé. Lorsque c'est fait, écrivez :

Code : Console - [Sélectionner](#)

```
cd <Emplacement de ndless>/src/arm/demo
```

N'oubliez pas : MSYS demande des slashes dans ce sens : '/'.
Remplacez <Emplacement de ndless> par l'emplacement de votre version de Ndless, comme par exemple C:/Users/rob36/Downloads/ndless/ndless_1.1.

Appuyez sur entrée puis écrivez :

Code : Console - [Sélectionner](#)

```
make
```

Cela devrait vous rappeler la compilation avec Qt si vous avez suivi un tutoriel concernant cette bibliothèque.

Après avoir appuyé sur entrée, si tout vos outils sont bien installés, vous devriez obtenir un message de ce type :

Code : Console - [Sélectionner](#)

```
nspire-gcc -Os -Wall -W -fpic -fno-merge-constants -D NON_CAS -c main.c
nspire-gcc -Os -Wall -W -fpic -fno-merge-constants -D NON_CAS -c utils.c
nspire-gcc -Os -Wall -W -fpic -fno-merge-constants -D NON_CAS -c vector.c
nspire-gcc -Os -Wall -W -fpic -fno-merge-constants -D NON_CAS -c particle.c
nspire-gcc -Os -Wall -W -fpic -fno-merge-constants -D NON_CAS -c particle_system.c
nspire-ld main.o utils.o vector.o particle.o particle_system.o gravity_particles.o -o demo
"/c/Program Files (x86)/yagarto/bin/arm-elf/objcopy.exe" -O binary demo.elf demo.bin
MakeTNS demo.bin
'demo.tns' successfully created!
rm -f demo.bin
mkdir -p ../../res/NON_CAS
mv demo.tns ../../res/NON_CAS
```

Si vous obtenez ce message : tous vos efforts vont enfin porter leur fruits !

Vos outils marchent et vous avez appris à compiler un programme.

Cette compilation a généré un fichier *demo.tns* dans le dossier *ndless\src\res\NON_CAS* que vous pourrez transférer sur votre calculatrice TI Nspire. Si vous possédez le modèle CAS, alors avant la compilation, ouvrez le fichier nommé *Makefile* dans le dossier de la demo et sur la première ligne, à la place de *NON_CAS* écrivez *CAS*. Le fichier *demo.tns* aura alors été généré dans le dossier *ndless\src\res\CAS*.

Programmer vos propres projets

Vous vous demander maintenant sûrement comment programmer votre super-jeu-de-la-mort-qui-tue .
Le plus simple et le plus pratique est d'ouvrir le fichier main.c dans le dossier demo et de tout effacer
puis de mettre ce code :

Code : C - [Sélectionner](#)

```
1 #include <os.h>
2 #include <common.h>
3 #include "utils.h"
4
5 asm(".string \"PRG\\\"\\n\");
6
7 int main(void)
8 {
9 showSimpleDialogBox("Hello Word", "Hello Word");
10 return 0;
11 }
```

Les trois `#include` sont indispensables ainsi que la ligne `asm(".string \"PRG\\\"\\n\");` pour chacun de vos programmes.

La fonction `showSimpleDialogBox()` permet d'afficher un boîte de dialogue. Le premier paramètre est le titre et le deuxième est le texte à l'intérieur.

Pour le moment je vous conseille de faire vos programmes dans le dossier demo. Pour connaître toutes les fonctions qui sont à votre disposition, vous devriez regarder dans les fichiers que l'on inclus au début du programmes. Ils contiennent les codes touches, les fonctions dont vous aurez besoins.

Pour vous aider, je peux vous indiquer `setPixel()` qui permet d'activer un pixel et `isKeyPressed` qui permet de vérifier si une touche est appuyée. Pour les paramètres je vous laisse vous débrouiller tout seuls. A vous de coder comme bon vous semble. Vous pourriez par exemple commencer par coder une fonction qui permet de dessiner une ligne à partir de `setPixel()`. Je peux également que vous pouvez utiliser 15 niveaux de gris.

Maintenant, je vous lâche dans la nature !

Au secours je ne peux pas me débrouiller tout seul, comment je dois faire ! A l'aide !

Bon , alors on va voir un exemples et faire un exercices.

Exemple

Nous allons voir l'utilisation de `setPixel()` et de `isKeyPressed()` en même temps.

Prenons ce code et analysons-le :

Code : C - [Sélectionner](#)

```
1 #include <os.h>
2 #include <common.h>
3 #include "utils.h"
4
5 asm(".string \"PRG\\\"\\n\");
6
7 int main(void)
8 {
9 clearScreen();
10 while(!isKeyPressed(KEY_NSPIRE_ESC))
11 {
12 setPixel(1, 1, 0);
13 }
14 return 0;
15 }
```

Lusqu'au début de la fonction main, normalement ça devrait aller si vous avez bien suivi.

Ensuite il y a la fonction clearScreen() qui permet, comme vous l'avez deviné en tant qu'anglophobe que grands adorateurs de la langue de Shakespeare ... d'effacer l'écran !

Plus tard vous avez la fameuse fonction setPixel(). Le premier paramètre est l'emplacement x du pixel que vous souhaitez modifier. Le deuxième est la position y du pixel et le dernier paramètre est la couleur soit le niveau de gris. Je vous rappelle qu'il y a 15 niveaux de gris.

Sachant que 0 est le noir et 15 le blanc, nous pouvons déduire que cette fonction va ici colorier le pixel en haut à gauche de l'écran en noir.

On peut voir que setPixel() est comprise dans une boucle while. La condition pour que cette boucle recommence est que la fonction isKeyPressed(KEY_NSPIRE_ESC) renvoie false.

Conclusion : ce code dessine un pixel noir tant que l'utilisateur n'a pas appuyé sur enter, et lorsque cela se produit, le programme se ferme.

Exercice

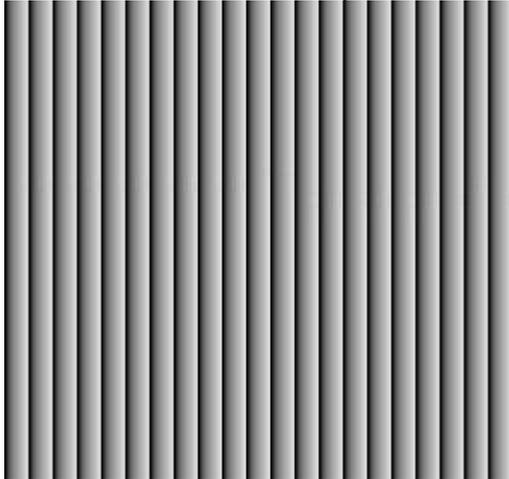
Il est temps que vous vous bougiez un peu maintenant . Comme disait ma prof de Français lorsqu'on écoutait pas :

Citation : Prof de Français

Il va falloir arrêter de faire la patate sur chaise !

Enfin bon passons cette citation.

Le but de cet exercice sera que vous obteniez ce résultat à l'écran :



Normalement, je n'ai pas besoin de vous dire quoi que ce soit si ce n'est que l'écran de la TI Nspire à une résolution de 320x240.

N'oubliez pas que vous programmez en C et que toutes les fonction habituelles sont disponibles !

Bonne chance !

...
...
...

Ca y est ? Super ! Je vais maintenant pouvoir vous donner mon code. Des personnes pourraient sûrement plus optimiser ce code mais je veux faire simple, donc voilà :

Code : C - [Sélectionner](#)

```
1 #include <os.h>
2 #include <common.h>
3 #include "utils.h"
4
5 asm(".string \"PRG\\\"\\n\");
6
```

```

7 int main(void)
8 {
9     int color, positionColonne, i;
10 clearScreen(); // J'efface l'écran
11 while(!isKeyPressed(KEY_NSPIRE_ESC)) // Je fais une boucle sinon le programme s'arrête
12 {
13     color=0; // On initialise la couleur
14     for(positionColonne=1;positionColonne<321;positionColonne++) // On l'initialise à 1
15     {
16         for(i=1;i<240;i++) // On trace le ligne
17             setPixel(positionColonne, i, color);
18
19         color++; // On change la couleur pour faire le dégradé
20         if(color>15) // Si la couleur est le blanc, alors on la remet en noir
21             color=0;
22     }
23 }
24 return 0; // Fin du programme
25 }

```

Voilà ! C'est maintenant pour nous vraiment le moment de nous quitter...

A bientôt !

RobFire