SimpleMenu LIBRARY

TUTORIEL

Que fait SimpleMenu?

Menu Library nous permet d'ajouter un menu à nos sketches. Les menus sont utilisés pour permettre à un usager de choisir quelles **actions** prendre et dans quel ordre il désire les exécuter.

Menu	→ Action ← Retour au menu	Actions

Quelles sont les étapes?

Pour utiliser un menu, nous devons avoir quatre dispositifs :

- Une librairie qui vous permet de déployer votre menu (SimpleMenu Library);
- Une manière de montrer le menu à l'usager (ECL…);
- Une manière d'utiliser le menu (habituellement un clavier);
- Une série de fonctions qui exécuteront les actions demandées

En résumé, nous indiquons quels sont les items de notre menu, nous préparons l'affichage, nous préparons le clavier tactile et nous préparons les actions

Préparation du menu

La préparation du menu est simple. Il suffit de le placer dans une chaîne de caractères, un item par ligne :

Exemple :

```
char items[] =
"-LIRE:000"
"--CAPTEUR:000"
"---CAPTEUR A1:101"
"---CAPTEUR A2:102"
"--TOUCHE:000"
"---BROCHE 4:103"
"---BROCHE 5:104"
"-ANGLE ROBOT:000"
"--BRAS:105"
"--BASE:106"
"-BOUGE ROBOT:107";

SimpleMenu monMenu(items); //Instanciation du menu
```

La première ligne déclare « items » de type char. Elle est égale à l'ensemble des lignes suivantes (jusqu'au point-virgule).

Chaque ligne suivante contient un item du menu, entre guillemets.

- Les tirets indiquent le niveau de l'item;
- L'intitulé de l'item (celui qui sera affiché);
- Un deux-points « : »;
- Un nombre entre "000" et "999" qui identifie l'**action** à accomplir si l'item est sélectionné.
  Il doit être d'exactement trois caractères et ne contenir que des chiffres.
  Si l'item a un sous-menu, on indique "000".

On ne doit pas oublier de terminer la dernière ligne avec un point-virgule.

Il ne reste qu'à instancier le menu.

Maintenant, SimpleMenu Library sait que :

The MENU PRINCIPAL est :
```
"-LIRE:000"
"-ANGLE ROBOT:000"
"-BOUGE ROBOT:107"
```

LIRE a un sous-menu :
```
"--CAPTEUR:000"
"--TOUCHE:000"
```

CAPTEUR a un sous-menu :
```
"---CAPTEUR A1:101"
"---CAPTEUR A2:102"
```

TOUCHE a un sous-menu :
```
"---BROCHE 4:103"
"---BROCHE 5:104"
```

ANGLE ROBOT a un sous-menu :
```
"--BRAS:105"
"--BASE:106"
```

Les items qui possèdent un sous-menu sont (numérotés 000) :
```
"-LIRE:000"
"--CAPTEURS:000"
"--TOUCHES:000"
"-ANGLE ROBOT:000"
```

Les items qui amorcent une action sont (numérotés de 101 à 107) :
```
"---CAPTEUR A1:101"
"---CAPTEUR A2:102"
"---BROCHE 4:103"
"---BROCHE 5:104"
"--BRAS:105"
"--BASE:106"
"-BOUGE ROBOT:107"
```

Préparation de l'affichage.

Nous devons nous référer aux consignes du fabricant de notre afficheur pour nous assurer que les connexions sont faites correctement. Habituellement, les Librairies fournissent des exemples de sketch (Comme « Hello World »). Nous devons nos assurer que ces sketches fonctionnent normalement avant de poursuivre notre sketch avec notre menu.

Exemple : la Librairie <LiquidCrystal.h> :

```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
#define LCD_COL 16
#define LCD_LINES 2
bool afficherDeNouveau = true;
```

Le drapeau « afficherDeNouveau » nous permettra de ne réafficher le menu que lorsque nécessaire.

Nous devons créer une routine permettant d'afficher le menu. Cette routine doit apparaître avant la section setup() de notre sketch.

En voici un exemple :

```
void afficheMenu() {
  if (afficher de nouveau()) {
    monLcd.clear();
    for (int i = 0 ; i < LCD_LINES ; i++) {
      monLcd.setCursor(0,i);
      monLcd.print(monMenu.getLine(i));
    }
    afficherDeNouveau = false;
  }
}
```

Dans la section setup() de notre sketch, il est possible d'avoir à insérer une ligne pour démarrer l'afficheur :

```
lcd.begin(LCD_COL, LCD_LINES);
```

Nous devons aussi insérer la ligne suivante pour que SimpleMenu connaisse le nombre de lignes de l'afficheur :

```
menu.setDisplay(LCD_LINES);
```

Préparation du clavier tactile.

Il existe une multitude de manières pour utiliser des interrupteurs avec un Arduino. Il suffit d'utiliser celui avec lequel nous sommes le plus à l'aise. Il est suggéré d'utiliser une solution anti-rebonds. De plus, il est préférable d'attendre que la touche ait été relâchée avant d'envoyer son code à SimpleMenu.

```
if (digitalRead(3) {while (digitalRead(3) {;} return 3;}
```

Nous devons indiquer quelles valeurs nous donnerons à SimpleMenu selon les quatre touches directionnelles (HAUT, BAS, GAUCHE, DROITE). Ces valeurs sont de type « byte », donc de 1 à 255. Habituellement nous réserverons 0 pour mentionner qu'aucune touche n'a été pressée. Dans la section setup() du sketch, nous allons ajouter :

```
monMenu.setKeypad(HAUT, BAS, GAUCHE, DROITE);
```

Ils doivent apparaître dans cet ordre.

Comment navigue-t-on dans le menu?

HAUT	Sélectionne l'item précédent, s'il existe
BAS	Sélectionne l'item suivant s'il existe
GAUCHE	Sélectionne le parent de l'item s'il existe
DROITE	Sélectionne le premier item de son sous-menu s'il en possède un. Sinon, le nombre qui est associé à l'item dans le menu sera retourné.

Dans la section loop() du sketch nous allons inscrire:

```
int action = monMenu.getAction(key);
```

Plus de details à la page **Erreur ! Signet non défini.**.

Préparation des actions

Les actions sont identifiées par un nombre dans le menu. Nous nous servirons de ce nombre dans une structure switch case :

```
void faire(int action) {
    case 101: { le code; break; }
    case 102: { le code; break; }
    case 103: { le code; break; }
    case 104: { le code; break; }
    case 105: { le code; break; }
    case 106: { le code; break; }
    case 107: { le code; break; }
  }
}
```

Afin de garder cette structure propre, nous pouvons remplacer « le code » par un appel à une fonction :

```
void make(int action) {
    case 101: { montreAnalogue(A0); break; }
    case 102: { montreAnalogue (A1); break; }
    case 103: { montreDigital(2); break; }
    case 104: { montreDigital (3); break; }
    case 105: { angleServo(1); break; }
    case 106: { angleServo(2); break }
    case 107: { bougeServos(); break; }
  }
}
```

Puis écrire le code dans ces fonctions.

Quelles sont les méthodes disponibles?

Méthodes obligatoires

Les méthodes suivantes doivent être appelées pour que SimpleMenu Library fonctionne.

void SimpleMenu::SimpleMenu(char* menuItems)

```
SimpleMenu monMenu(items);
```

Cette méthode permet d'instancier un objet SimpleMenu. Son appel doit être fait avant la section setup() de notre sketch.
SimpleMenu est le nom du constructeur de l'objet;
monMenu est le nom que nous donnons à l'objet;
items est le nom que nous avons donné à la chaîne contenant notre menu.

void SimpleMenu::setDisplayLines(byte displayLines)

```
monMenu.setDisplayLines(LCD_LINES);
```

Cette méthode permet d'indiquer à SimpleMenu Library le nombre de lignes de notre afficheur. Son appel doit être fait dans la section setup() de notre sketch.
displayLines est le nombre de lignes de votre afficheur.

void SimpleMenu::setKeypad(byte UP, byte DOWN, byte LEFT, byte RIGHT)

```
monMenu.setKeypad(HAUT, BAS, GAUCHE, DROIT); // Si vous avez utilisé « #define » ou…
monMenu.setKeypad(1, 2, 3, 4);
```

Cette méthode permet d'indiquer quelles valeurs correspondent aux touches directionnelles que nous allons transmettre à SimpleMenu Library. Son appel doit être fait dans la section setup() de notre sketch. Il est essentiel de les énumérer dans cet ordre : HAUT, BAS, GAUCHE, DROIT.

Méthode reliée à l'affichage

char* SimpleMenu::getLine(byte line)

```
monMenu.getLine(0);
```

Cette méthode retourne le libellé de l'item du menu qui doit apparaître à la ligne « line » de l'afficheur. On l'utilise à l'intérieur d'une boucle « for » pour afficher toutes les lignes du menu. L'item courant est précédé du signe « plus grand que ». Les autres sont précédés d'une espace.

```
monLcd.clear();                                  //Vider l'afficheur
for (int i = 0 ; i < LCD_LINES ; i++) {          //Pour chaque ligne
  monLcd.setCursor(0,i);                           //Placer le curseur au début de la ligne
  monLcd.print(monMenu.getLine(i));                //Afficher l'item correspondant
}
```

Méthode reliée à la gestion du menu

int SimpleMenu::getAction(byte key)

```
int action = monMenu.getAction(touche);
```

Quand cette ligne est exécutée il se passe trois choses :

- Nous indiquons quelle touche a été activée;
- SimpleMenu détermine quel est le nouvel item du menu qui devient actif;
- SimpleMenu retourne l'action attachée à l'item du menu (ou 0 s'il n'y a pas d'action à accomplir)

Comprendre la section loop()

```
void loop() {
  if (afficherDeNouveau) {                //Si nous devons réafficher
    afficheMenu();                          //Afficher
  }
  byte touche = lireBoutons();            //Lire les quatre boutons
  if (touche > 0) {                       //Si une touche a été pressée
    int action = menu.getAction(touche);    //Remettre la touche à SimpleMenu
    if (action > 0) {                       //Si une action a été retournée
      faire(action);                        //Accomplir l'action
    }
    afficherDeNouveau = true;             //Nos devons réafficher
  }
}
```

Mise en garde

Afin de réduire au minimum la quantité de mémoire occupée par les variables du sketch, SimpleMenu ne réquisitionne que la mémoire nécessaire pour le nombre d'items contenus dans le menu.

Quand l'IDE de l'Arduino termine la compilation, il indique la quantité de mémoire utilisée pour les variables (en octets et en pourcentage). Cette quantité n'inclut pas la mémoire supplémentaire réquisitionnée par SimpleMenu, car celle-ci est allouée après la compilation. (Allocation de mémoire dynamique).

SimpleMenu réquisitionne 7 octets par item du menu, plus 7 autres octetc pour usage interne. Ainsi, pour un menu comportant 10 items, la mémoire dynamique réquisitionnée et de 7*10+7=77 octets.

J'espère que SimpleMenu vous sera utile dans vos projets.
Jacques Bellavance